Departamento de Ingeniería Telemática

PROYECTO FIN DE CARRERA

Algoritmos y videojuegos

Autor: Mª Carmen Martínez Ferrari

Tutor: José Jesús García Rueda

Leganés, marzo de 2012

Universidad Carlos III de Madrid

Ingeniería Técnica de Telecomunicaciones: Sistemas de Telecomunicaciones

Título: Algoritmos y videojuegos.

Autor: María del Carmen Martínez Ferrari.

Tutor: José Jesús García Rueda.

EL TRIBUNAL

Presidente: Pedro J. Muñoz Merino

Secretario: Roberto González Sánchez

Vocal: Matilde Pilar Sánchez Fernández

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día 30 de Marzo de 2012 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de:

RESUMEN

El presente proyecto explica el diseño y creación de un videojuego educativo que fusiona dos estilos de juegos muy diferentes, un estilo clásico y uno moderno, las aventuras gráfico-conversacionales y los juegos de realidad alternativa, los cuales, en los últimos años están teniendo un gran auge no solo dentro del mundo de los videojuegos, sino también en el ámbito educativo.

El objetivo de este trabajo es aportar las habilidades mentales necesarias en la resolución de cierto tipo de problemas, ya que según varios estudios realizados, ciertos problemas de lógica facilitarán el posterior aprendizaje de la algoritmia a aquellos estudiantes que se introduzcan por primera vez en el mundo de la programación.

Es debido a la dificultad que conlleva dicho aprendizaje por la que se realiza este videojuego, gracias al cual, el usuario podrá adquirir dichas habilidades mediante el entretenimiento, la diversión y la intriga que crea formar parte de una misión de rescate.

ABSTRACT

This project explains the design and creation of a game. This game mixes two styles very different, a classical conversational and graphic adventure and a new one, the alternate reality game. Nowadays, the last ones have a huge success in game world and in an educational environment.

The aim of this project consists in giving the mind skills to resolve problems. Some studies have shown that logical problems can make easy the future learning of algorithms to students who never had contact with programming.

This videogame is made because the effort of learning this kind of skills. The user will learn these sorts of skills in a funny way and will be intrigued by a ransom mission.

Agradezco a todas las personas que me han apoyado en todos los momentos, a mi familia, a mis amigos y a todas las personas que directa o indirectamente han contribuido a este momento.

1. INTRODUCCIÓN

1.1. Introducción y contexto

1.2. Motivación

1.3. Objetivos

1.4. Metodología de trabajo

1.5. Organización del documento

1.1. Introducción y contexto

Un videojuego es un software creado para el entretenimiento y está basado en la interacción entre una o varias personas y se ejecuta en un PC, una videoconsola o un teléfono móvil.

Por lo general, los videojuegos hacen uso de otras maneras de proveer la interactividad e información al jugador. El audio es casi universal, usándose dispositivos de reproducción de sonido, tales como altavoces y auriculares. En otros se produce una vibración cuando se intenta simular fuerza.

Hoy en día tal es la variedad de videojuegos, aventuras, lucha, disparos, deportes o rol, que se han comenzado a usar como medio para la educación y aprendizaje en niños e incluso de personas adultas.

En la actualidad, los chicos de unos 15 años (pertenecientes a la Generación Interactiva, GI) están creciendo en un contexto social, cultural y educativo radicalmente distinto al que acompañó a sus padres y profesores. Los "viejos" modelos educativos ya no se ajustan de forma conveniente a su forma de ser, lo que origina tensiones.

El adjetivo "interactiva" se refiere a que existe una acción recíproca entre las tecnologías y las personas, de tal manera que éstas y aquéllas se influencian mutuamente. Por ejemplo, a la vez que la tecnología cambia al ser humano, este también está continuamente configurándola de maneras no previstas inicialmente, inventando nuevos usos o ideando las más insospechadas aplicaciones.

Los chicos de esta generación se mueven en un medio en el que la información se organiza de una manera muy distinta a la utilizada en la escritura convencional, es por ello que a algunos los libros les pueden resultar tan extraños como los hipertextos a los adultos. Les gusta la fantasía, lo reflejan los libros que leen, las películas que ven, los videojuegos que juegan y el imparable éxito de los mundos virtuales que nacen en cualquier esquina del ciberespacio. Su identidad digital tiene tanto valor o más que la real. Por ello le conceden tanta importancia y le dedican tanto tiempo y atención a lo que se dice de ellos en la red. Además, experimentan nuevas formas de relacionarse y, en muchas ocasiones, expresan emociones y proporcionan información propia o de otras personas en diversos formatos, como texto, audio o vídeo.

1.2. Motivación

El presente proyecto ha nacido de la idea de facilitar el aprendizaje de la lógica necesaria en la programación. Como todo comienzo no suele ser sencillo, aprender a programar tampoco lo es, por lo que la existencia de un software que sirva para facilitar la entrada en el mundo de la programación es bastante útil y necesaria.

Programar no es solo escribir un código en un lenguaje en concreto, primero hay que analizar el problema y posteriormente diseñar una secuencia de pasos finita para

poder resolverlo, lo que se conoce como algoritmo. Una vez diseñado el algoritmo se pasará al lenguaje correspondiente en la aplicación.

Ya que muchas personas consideran que aprender a programar es complejo y muchas desisten de intentarlo, vamos a diseñar un juego que facilite su aprendizaje.

Muchos expertos en el área de la programación han afirmado que para realizar en un ordenador procedimientos que realicen determinadas tareas se deben seguir cuatro etapas: [1]

1. Analizar el problema
2. Diseñar un algoritmo
3. Traducir el algoritmo a un lenguaje de programación
4. Depurar el programa

Por tanto, la idea es realizar un juego que sea educativo y que sirva para facilitar el posterior aprendizaje de la algoritmia a la persona que juegue con él. No se debe olvidar que ante todo es un juego y debe tener la capacidad de entretener y crear adicción al jugador a la vez que aprende.

El juego a realizar es un juego que mezcla varios estilos, por un lado es de tipo aventura conversacional (incluyendo imágenes, música y vídeos) y tipo de realidad alternativa, estilo muy utilizado en publicidad, en el que los jugadores deben lograr el objetivo planteado en un comienzo, siendo necesario resolver pruebas y acertijos en su camino.

La principal motivación de este proyecto es el diseño y desarrollo de una aplicación software, que en este caso es un juego, que permita a sus jugadores ir "entrenando su cabeza" para abordar un determinado tipo de problema en el mundo de la algoritmia y/o programación de una forma divertida a la vez que se aprende, ya que no tiene por qué ser un juego aburrido si es educativo, solo hay que dar la proporción justa de educación y adornarla de tal manera que el jugador no lo vea como algo que hay que aprender, sino como un reto que hay que superar para poder avanzar en el juego y conseguir el objetivo marcado desde el principio.[10]

Dado el tipo de juego, la interfaz de usuario es sencilla y fácil de utilizar, ya que es similar a las interfaces usadas en las aventuras gráficas (aunque no se puede considerar una aventura gráfica, ya que realizar un videojuego de ese estilo necesitaría demasiados recursos gráficos, se perdería tiempo en su búsqueda y no aportaría demasiado aprendizaje) en las que se pueden ver la sala en la que se encuentra el personaje, las personas con las que hablar, los objetos que mirar, coger o usar.

En definitiva, un trabajo que puede ser de utilidad no solo a los alumnos de carreras técnicas, sino a todos que deseen comenzar en el estudio de la algoritmia.

1.3. Objetivos

Los objetivos planteados en este proyecto se centran en el aprendizaje por descubrimiento, en el que el sujeto, en lugar de recibir los contenidos de forma pasiva, descubre las habilidades y las formas de resolver problemas para adaptarlos a su esquema cognitivo.

En primer lugar, para entender el objetivo primordial hay que saber qué es un algoritmo: un algoritmo es un conjunto de instrucciones o reglas definidas, ordenadas y finitas que permite realizar una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad.

Por lo tanto, el principal objetivo se trata de lograr mediante el diseño de un juego didáctico, hacer posible que el estudiante alcance unas habilidades intuitivas de resolución de problemas mediante algoritmos, ya que hasta el momento actual, y seguirá siéndolo, el conocimiento de algoritmos computacionales es vital para el desarrollo de aplicaciones para ordenadores y el manejo y dominio de la lógica de programación para resolver problemas.

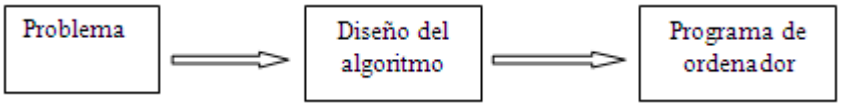

Ilustración 1 Pasos previos a la programación

Para la resolución de algoritmos existen diversas técnicas algorítmicas que se aplican dependiendo del problema en cuestión, como son por ejemplo la famosa técnica de “divide y vencerás”, que consiste en dividir un problema complicado en varios problemas más sencillos con el objetivo de resolver el problema inicial. En apartados posteriores nos centraremos en varias técnicas algorítmicas. El diseño se ha realizado de manera que pueda ser utilizado como base para cualquiera de las asignaturas introductorias de la algoritmia.

No tenemos que olvidar que a pesar de todo, estamos ante un juego y por lo tanto debe llamar la atención al jugador y divertirle [34]. Nuestro juego constará de una narración por parte de un personaje del juego que atraiga y que involucre al jugador en la historia, de esa manera nos aseguraremos de que jugará hasta el final. En el juego se deberán resolver pruebas, y solo si se resuelven de forma correcta, se podrá avanzar en la historia, por tanto, aprenderá a solucionar problemas de forma algorítmica.

Introduciremos al jugador en una misión, el rescate de Lidia, la cual fue secuestrada y no tenemos pistas sobre su desaparición, solo una nota y unas pinturas hechas en la pared de su habitación. Mediante el uso de vídeos, recepción de emails en la cuenta de correo del jugador y el uso de imágenes de personas reales se intenta simular una realidad alternativa, cuyo fin es confundir al jugador, hacer que la persona que está jugando al videojuego llegue a pensar que no es un juego y que está verdaderamente en una misión de rescate.

Para ello, se ha recurrido a un estilo nuevo de juego, fusionando dos estilos muy distintos, por un lado el juego usa el estilo de un *Alternate Reality Game* (ARG) para crear la atmósfera de que lo que está sucediendo es todo real y el de la aventura conversacional para crear la parte visual propia de todo videojuego. Para poder hacer esta fusión, son necesarios dos protagonistas, el primero, el personaje virtual, que pide ayuda en la búsqueda, que se moverá por los escenarios virtuales, haciendo parecer que son de verdad, y el segundo, y más importante, el propio usuario sentado delante del ordenador con la misión de ayudar en la búsqueda, de resolver los problemas planteados que el jugador virtual es incapaz de resolver ya que va pidiendo ayuda constante durante el juego y enviando al jugador las pistas que va encontrando en su camino.

1.4. Metodología de trabajo

En este proyecto se han seguido una serie de fases en su realización.

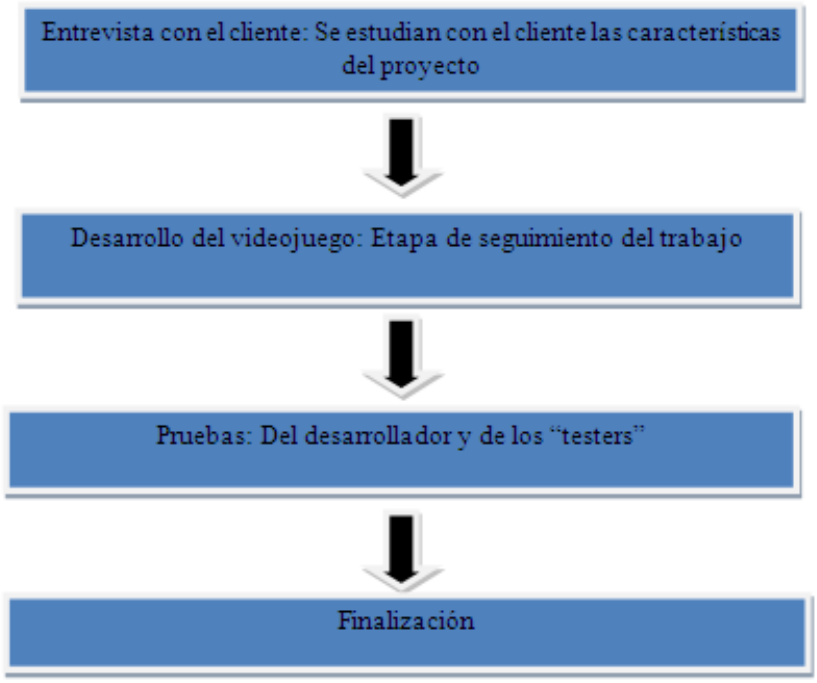

Ilustración 2 Metodología de trabajo

a) Entrevista con el cliente

En esta fase el cliente, en este caso el tutor del proyecto, determina las características generales y los requisitos del proyecto, como por ejemplo, el tipo de videojuego que quiere. El cliente pide al desarrollador, el proyectando, un software, dejando a libertad del desarrollador cierto tipo de elecciones siempre y cuando se cumplan los requisitos pedidos. En nuestro caso, el cliente pidió un juego en el cual se fuera aprendiendo el pensamiento algorítmico en los esquemas mentales de los estudiantes, pero no especificó los algoritmos a

aprender ni los juegos de lógica a superar durante el juego, eso se dejó al desarrollador.

Tampoco exigió una interfaz gráfica concreta, solo alguna similar usada en este tipo de juegos. Se llegó a la conclusión de que con introducir una pequeña cantidad de imágenes en el juego era suficiente, ya que las aventuras gráficas, como su nombre indica, tienen demasiadas imágenes, eso en este caso sería muy laborioso y aportaría poco aprendizaje, por lo que se llegó a dicho acuerdo con el cliente.

Otro requisito impuesto por el cliente fue que el juego tuviera un buen argumento, que gustase a todo tipo de jugadores, que introdujera cierto misterio para lograr que el jugador no se cansase de jugar.

b) Desarrollo del videojuego

Una vez llegados a esta etapa, el desarrollador ya tiene una idea de lo que quiere el cliente y se comienza a programar el juego.

Tras la primera entrevista hubo una etapa de documentación. Era necesario investigar sobre el tema de la algoritmia y los videojuegos. Existen multitud de libros y programas que enseñan algoritmia, pero no mediante juegos de ordenador. Existen problemas de lógica, que pueden resolverse mediante algoritmos, pero desde un punto de vista académico.

Para el desarrollo del videojuego, ha sido necesario repasar la programación en Java, profundizar en Swing y aprender otras librerías, como son JDOM, relacionado con el lenguaje XML, siglas en inglés de *eXtensible Markup Language*, para almacenar la información de cada escena del videojuego, JavaMail para enviar emails al jugador durante la ejecución del juego que sirven de pistas para el avance en el mismo y Java Media FrameWork para reproducir sonido y vídeo que hacen más entretenido y más atrayente el juego para el usuario.

Para enlazar todo, ha sido necesario estudiar qué es un motor de juego. Para ello, se ha recurrido a un libro para niños en el que se explica la filosofía de una aventura conversacional, cómo diseñarla y, en esencia, cómo realizar un motor de juego. [2]

También se estudió la reflexión en Java para la creación de objetos sin conocer de antemano la clase a la que pertenecen, ya que el motor de juego va leyendo de la base de datos el nombre del mini-juego a ejecutar.

Esta etapa es una etapa de desarrollo y seguimiento, ya que el cliente y el desarrollador van concertando entrevistas en las cuales se muestra al cliente lo realizado y si éste está de acuerdo, se sigue con el trabajo por ese camino. Si no

lo está, se hacen las correcciones oportunas, buscando en todo caso la satisfacción del cliente.

En un principio el cliente no veía muy claramente que el juego fuera un ARG, ya que no es un ARG multijugador, y en cierto modo, parecía una aventura conversacional normal. Para mejorar en este aspecto, se decidió usar fotografías reales, y no dibujos o pinturas, con el fin de dar apariencia de realidad. Por ello que los personajes son fotografías de personas y se han seleccionado imágenes de lugares reales como museos, aeropuertos y edificios.

Por otro lado, también se ha intentado conseguir la atmósfera de un ARG mediante el envío (personalizado al usuario, ya que el juego pide la introducción de nombre de usuario y dirección de correo electrónico) de emails por parte de Saúl, protagonista de la aventura.

c) Fase de pruebas

Para determinar el nivel de calidad se deben efectuar unas medidas o pruebas que permitan comprobar el grado de cumplimiento respecto de las especificaciones iniciales del sistema, es decir, una vez completado el desarrollo del videojuego, es hora de pasarle las pruebas para ver si funciona.

La fase de pruebas, en proyectos reales, necesita de profesionales del sector que estén capacitados en lenguajes de desarrollo, técnicas de pruebas y en las herramientas necesarias para realizar dichas pruebas. Incluso el conocimiento que debe tener un ingeniero de prueba es, en muchas ocasiones, mayor al del desarrollador del software.

Las pruebas de software se integran dentro de las diferentes fases del ciclo del software dentro de la Ingeniería de software. La fase de pruebas es necesaria para identificar posibles fallos en la implementación, en la calidad o usabilidad del programa, en este caso el videojuego.

En general, los programadores distinguen entre errores de la propia programación, denominados también *bugs*, fallos en la semántica de un programa, y defectos de forma, en los cuales el programa no realiza lo que el usuario espera.

Como en todos los videojuegos, las primeras personas en probar su funcionamiento son los propios desarrolladores. En este caso, es el proyectando quien se ocupa de esa labor.

La fase de pruebas se ha ido realizando de forma modular a lo largo de toda la programación del videojuego. Cada elemento añadido era probado. Por ejemplo, se añadía un método y se comprobaba si realizaba su función de forma correcta.

Una vez realizadas estas pruebas, denominadas de caja blanca, se pasaba a comprobar el funcionamiento del módulo en cuestión. Se realizaron pruebas semejantes a las pruebas de caja negra, es decir, en ellas se comprobaba la integración de los subsistemas.

Llegados a un punto avanzado del proyecto, se establecía una reunión con el cliente para mostrar la situación actual del proyecto, en la cual, se comentaban los cambios necesarios a realizar, tanto por nuevos fallos encontrados en la ejecución del programa, como cambios de diseño que hicieran la interfaz más intuitiva y práctica.

Por último, se pasó el programa a los denominados *testers*, amigos del proyectando que ponían al juego en situaciones límite, con el fin de buscar errores que pudieran estar ocultos tras posibles combinaciones de opciones que un jugador podría pensar a lo largo del juego.

El proceso de pruebas es fundamental a la hora de buscar fallos. Hoy en día es vital evaluar la calidad de todo producto construido, ya que cuanto antes se observe un fallo, menor coste tendrá.

d) Finalización

Tras la continua relación con el cliente, no se considera el proyecto acabado hasta su total satisfacción.

Tras esta fase, se fabrica el videojuego y se realiza la redacción de la memoria que acompaña al software. Posteriormente se realiza la defensa pública ante un tribunal.

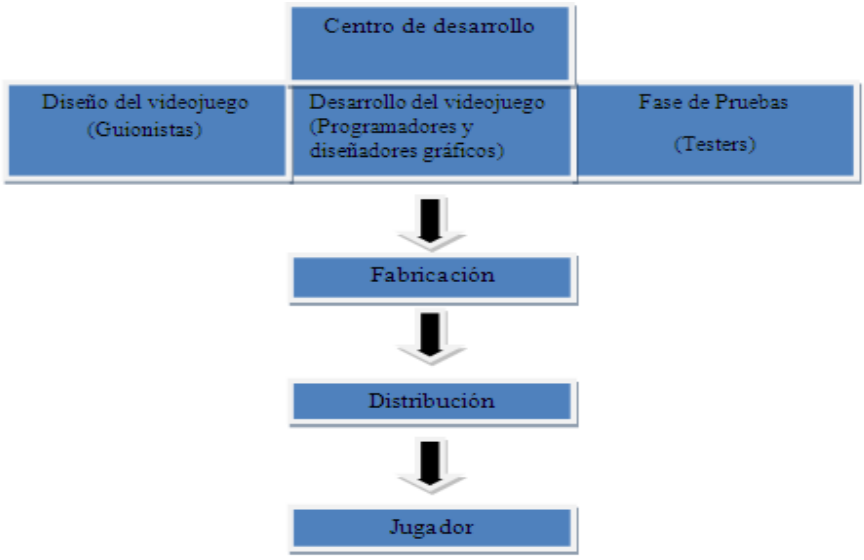

Ilustración 3 Fases de creación de un videojuego

1.5. Organización del documento

La memoria del proyecto está formada de los siguientes capítulos:

1. **Introducción**

   En esta sección se da una idea global del proyecto. En qué consiste, su principal motivación y objetivos.

2. **Estado del arte**

   Capítulo en el cual se pretende realizar un estudio sobre la problemática que tiene todo aquél que se enfrenta al aprendizaje de la programación por primera vez. Incluye estudios realizados, el uso de los videojuegos en la enseñanza y un estudio de dos estilos muy distintos de juego, las aventuras gráfico-conversacionales y los juegos de realidad alternativa.

3. **Desarrollo del proyecto**

   Es un capítulo en el que se van a comentar las principales decisiones de diseño que se han tenido que tomar, la arquitectura adoptada para la realización del videojuego, requisitos de partida, etc. Para ello, en cada decisión de diseño se explica el por qué de esa elección y la/s alternativa/s existente/s sobre las que se puede decidir, argumentando finalmente la elección utilizada. Sobre la arquitectura se detallarán las ventajas de la utilización de ésta y se explicarán en profundidad cada uno de los niveles por los que está formada.

4. **Conclusiones y trabajos futuros**

   Es un capítulo que va destinado a la exposición de las conclusiones obtenidas tras la realización del proyecto, y a su vez, se mostrarán una serie de mejoras y trabajos futuros que pueden ser implementadas en otro proyecto posterior para mejorarlo.

2. ESTADO DEL ARTE

2.1. Historia de los videojuegos

2.2. Estudio del empleo de los videojuegos en la enseñanza

2.3. Juegos de Realidad Alternativa (ARGs)

2.4. Aventuras gráficas con usos educativos

2.5. ¿Qué es un algoritmo?

2.6. Experimentos previos

2.7. Técnicas algorítmicas

2.8. Conclusiones

2.1. Historia de los videojuegos

La historia de los videojuegos comenzó en 1947, cuando la idea de un videojuego fue concebida y patentada por Thomas T. Goldsmith Jr y Estle Ray Mann. El juego consistía en el lanzamiento de misiles contra un objetivo. [11]

En 1952 Alexander Sandy Douglas creó el juego "Las tres en raya", también conocido como "OXO". Fue el primer juego en tener una versión digital. En 1958 William Higinbotham hizo el conocido juego "Tennis for two". Consistía en una línea horizontal que representaba el suelo del campo de tenis y de una pequeña línea vertical en la mitad del campo que representaba la red. Los jugadores debían elegir el ángulo en que debía salir la bola y golpearla.

Los años 80 comenzaron con un fuerte crecimiento en el sector del videojuego promovido por la popularidad de las salas recreativas y de las primeras videoconsolas aparecidas durante la década de los 70.

Sin embargo, en 1983 comenzó una crisis, sobretodo afectó a EEUU y Canadá, lo que conllevó a que el mundo de los videojuegos no evolucionara demasiado. A la salida de dicha crisis, la mayoría de los juegos solo contenían unas pocas pantallas que se repetían en un bucle y el objetivo simplemente era lograr la máxima puntuación.

Nintendo introdujo algo nuevo, ya que por primera vez se tenía un objetivo y un final en un videojuego. En los años posteriores otras compañías emularon su estilo de juego y a partir de entonces, gracias a la evolución de la tecnología y la aparición de nuevas plataformas, el mundo del videojuego siguió creciendo con más velocidad hasta la actualidad.

En 1976 apareció "Colossal Cave Adventure de Willie Crowther y Don Woods", que dio origen a las aventuras conversacionales, una opción diferente al resto de juegos basados en reflejos. Este tipo de juegos se basaban en la toma de decisiones, solución de acertijos, y elección de caminos, para llegar a un final. Estos juegos fueron el origen de las aventuras gráficas y novelas interactivas.

Durante el año 2008 en España se gastaron mil cuatrocientos treinta y dos millones de euros en productos relacionados con la industria del videojuego [4], setecientos cuarenta y cuatro en software, es decir en juegos y seiscientos ochenta y ocho en hardware, es decir, en aparatos para jugar. Estas cifras sitúan a España en el cuarto lugar de Europa, por detrás del Reino Unido, Francia y Alemania.

La industria del videojuego es la que más ingresos genera de todas las relacionadas con el ocio audiovisual e interactivo en España. Así, en 2008, frente a los más de mil cuatrocientos euros invertidos en adquirir productos relacionados con los videojuegos, se gastaron doscientos cuarenta y cinco en películas en DVD o doscientos doce millones en música. Es decir, por cada euro invertido en ocio audiovisual, cincuenta y siete céntimos fueron en artículos vinculados al videojuego. [9]

2.2. Estudio del empleo de los videojuegos en la enseñanza

La siguiente tabla muestra una estadística que relaciona el mundo del videojuego y los chicos de 15 años:

Actividad	Chicos de 15 años	Chicas de 15 años	Total
Tiene videoconsola	96%	88,3%	93,6%
Juega con el móvil	44,4%	33,1%	39,4%
Visita en internet lugares relacionados con juegos	76,4%	46,5%	63,4%
Juego en la red	71,4%	45,5%	60,2%

Tabla 1 Nativos Interactivos. Los adolescentes y sus pantallas: reflexiones educativas

Por estas estadísticas y por estudios relacionados en el aprendizaje, nos vamos a centrar en los juegos como actividad educativa. El juego es una actividad divertida, a través de la cual se puede desarrollar integralmente la personalidad del hombre, y en particular, su capacidad creadora. Mediante el juego, los chicos pueden investigar sobre nuevas formas para explorar la realidad, se puede desarrollar la imaginación, pensar en varias alternativas en un problema, desarrollar varios estilos de pensar, favoreciendo incluso el trato con otras personas si se juega en grupo.

Si se usaran nuevas formas de enseñanza en clase, como por ejemplo, usando juegos didácticos, se rompería el formalismo, se podría lograr una alta participación del alumno y como consecuencia, mejores resultados en el aprendizaje de la materia, ya que:

- ✓ Aumentaría el número de alumnos en clase, ya que estarían motivados.
- ✓ Al estar más motivados, aumentaría el interés en la materia y se profundizaría en el estudio.
- ✓ Se interiorizarían los conocimientos por medio de la repetición sistemática, dinámica y variada.
- ✓ Se aumentaría el trabajo en equipo ya que se podrían crear grupos de jugadores y se involucrarían en buscar la solución en conjunto.
- ✓ Lograría responsabilidad y compromiso con los resultados del juego, por lo que cada individuo se preocuparía por aportar su grano de arena y se fomentaría el estudio individual.

Aunque para que todo esto tenga éxito entre los alumnos, no se pueden realizar juegos de cualquier forma, los juegos didácticos deben tener unos objetivos definidos, unos contenidos concretos y los resultados de los alumnos deben ser evaluables.

Los elementos necesarios para lograr el éxito en este tipo de juegos:

- ✓ Delimitación clara y precisa del objetivo que se persigue con el juego.
- ✓ Metodología a seguir con el juego en cuestión.
- ✓ Instrumentos, materiales y medios que se utilizarán.
- ✓ Roles, funciones y responsabilidades de cada participante en el juego.
- ✓ Tiempo necesario para desarrollar el juego.
- ✓ Reglas que se tendrán en cuenta durante el desarrollo del juego.
- ✓ Lograr un clima psicológico adecuado durante el desarrollo del juego.
- ✓ Papel dirigente del profesor en la organización, desarrollo y evaluación de la actividad.
- ✓ Acostumbrar a los estudiantes a escuchar y leer.

Como hemos visto, los productos tradicionales van perdiendo terreno, en prácticamente todos los sentidos, y los videojuegos tienen un peso muy importante en la cultura del entretenimiento. Por eso, la creación de un videojuego educativo es símbolo de diversión aunque haya un aprendizaje de trasfondo. Además, los usuarios pueden desarrollar ciertas habilidades mientras juegan a un videojuego: [5]

1. **Destrezas motoras**: ante un juego virtual los usuarios pueden probar y practicar nuevos movimientos de actuación que les dan uniformidad y regularidad en su manera de proceder derivada de su experiencia de juego.
2. **Información verbal**: los juegos incluyen un amplio contenido verbal y escrito que el usuario tiene la obligación de comprender y asimilar para continuar con la partida.
3. **Destrezas intelectuales:** los jugadores aprenden a conectar la información dada y crean redes de significados. Los juegos creativos permiten desarrollar la creatividad y bien concebidos y organizados propician el desarrollo del grupo a niveles creativos superiores. También desarrollan la creación de nuevas ideas para resolver problemas de la vida real.
4. **Actitudes**: se enseña la moralidad a la hora de jugar, el espíritu de solidaridad, dar y recibir ayuda, el espíritu crítico y autocrítico, la tenacidad, la perseverancia, la responsabilidad, la audacia, la sistematicidad, el compañerismo, la cooperación, la lealtad, la seguridad en sí mismo.
5. **Estrategias cognoscitivas**: potencian una mayor atención/concentración en la lectura, por ejemplo. También se fomentan las capacidades lógicas, la fantasía, la imaginación, la iniciativa, la investigación científica, los conocimientos, las habilidades, los hábitos, el potencial creador, etc.

Características del juego	Valores adquiridos
Diversión	Satisfación
Jugar	Inmersión
Normas y reglas	Estructura
Metas y objetivos	Motivación
Interacción con el juego	Saber hacer
Reciprocidad	Aprendizaje con respuesta inmediata
Adaptación	Carácter abierto
Ganar la partida	Gratificación del ego, auto-superación
Competitividad, obstáculos	Adrenalina
Resolver problemas	Fomento de la creatividad
Interacción social	Aprendizaje social
Narratividad	Emociones

Ilustración 4 Características de cómo los videojuegos contribuyen a la adquisición de aprendizaje de los usuarios.

Se han realizado estudios sobre los juegos concretos de lógica, cuyo fin es conocer si la realización de ciertas actividades, como resolución de puzles y acertijos ayuda a mejorar estos aspectos. Algunos ejemplos son:

a) Puzles de alambres

Estos puzles plantean verdaderos retos, a través de actividades lúdicas que despiertan la curiosidad y el interés de la persona que los está usando, haciéndolos propensos a explorar algunas cualidades del espacio tridimensional. Es por ello que son propicios emplearlos en la enseñanza de las matemáticas ya que experiencias protagonizadas por niños, muestran que es posible introducirlos, que llegan a motivarles y les introduce en la resolución de problemas en la vida real.

La resolución, el análisis y el estudio de los puzles requiere apreciar aspectos topológicos, como son huecos, posiciones, enlaces y aspectos geométricos, formas y distancias, por lo que se considera su resolución útil en el campo de las matemáticas ya que pone a los alumnos en situación de afrontar problemas reales de naturaleza geométrica, que exigen el estudio de las formas, medidas y disposiciones topológicas, aunque no está muy extendido entre los profesores.

En el currículo español de matemáticas para la Educación Primaria, niños de edades comprendidas entre 6 y 12 años, se insiste en aspectos del estudio métrico y geométrico que pueden abordarse con tareas lúdicas relacionadas con el análisis, clasificación y resolución de puzles de alambre. En el currículo andaluz para el mismo nivel (Junta de Andalucía, 1992), se alude específicamente a que se propongan actividades que favorezcan en el niño la interiorización consciente de

dimensiones topológicas, y a ello puede colaborar de manera evidente trabajar en el aula de matemáticas con puzles de alambre.[4]

b) Juegos de acertijos como el famoso juego Profesor Layton

Según estudios los juegos en los que hay que resolver acertijos pueden fortalecer el cerebro y prepararlo para la actividad cognitiva. (El concepto de cognición hace referencia a la facultad de los seres de procesar información a partir de la percepción, el conocimiento adquirido (experiencia) y características subjetivas que permiten valorar la información).

Por medio de un análisis de registros MRI, resonancias magnéticas del cerebro, se estudiaron unas imágenes que han permitido aplicar cuatro métodos [5]:

1. Un análisis de **volumetría** en que se puede observar el aumento y la reducción regional de materia gris que promueve la potencia del cerebro para procesar información.
2. Un análisis de **área de superficie**, con el que se presentan mayores cambios en zonas relacionadas con procesos cognitivos, tales como razonamiento, planificación, lenguaje, cálculo y procesamiento espacial y visual.
3. Análisis de **difusión** que permite observar las conexiones entre diferentes zonas del cerebro. Fortalece ciertas conexiones y además se podría prevenir el deterioro de otras.
4. Análisis de conectividad en reposo.

Los jugadores mejoraron en todos estos análisis y aunque los resultados son preliminares, se fortalecieron la compresión verbal, el procesamiento viso-espacial, la percepción visual, la orientación topográfica, la planificación y razonamiento, que coinciden con los aspectos más necesarios para resolver problemas propuestos en el juego del Profesor Layton.

c) Brain Training

Es un videojuego de lógica y puzles. El jugador se encuentra el doctor Kawahima, su creador, quien va dándole consejos y explicándole la función y el desarrollo de cada una de las pruebas a las que debe enfrentarse. Suelen ser ejercicios variados en los que existen problemas de cálculo y retención de datos, de lectura y ortografía, sudokus, etc., todo ello con el fin de ejercitar la mente. Al principio solo es posible realizar unos pocos ejercicios, pero con el avance se van desbloqueando otros nuevos. Permite realizar una prueba compuesta por varios ejercicios disponibles para determinar la “edad cerebral” del jugador. También consta de unos gráficos para consultar los datos de los resultados de los distintos ejercicios realizados, pudiendo apreciar la evolución del jugador.

Hasta aquí hemos podido comprobar las ventajas de aprender jugando, ¿Por qué no aprender a pensar de forma algorítmica mientras nos divertimos? La algoritmia es uno de los pilares de la programación y su relevancia se muestra en el desarrollo de

cualquier aplicación, más allá de la mera construcción de programas. Es por ello por lo se ha querido diseñar un software que se introduzca al jugador en el mundo de los algoritmos de una forma divertida.

2.3. Juegos de Realidad Alternativa

2.3.1. Introducción

ARGs, cuyas siglas significan *Alternate Reality Game*, son juegos interactivos que usan el mundo real como soporte para contar una historia. Los ARGs utilizan diferentes elementos que ayudan a contar la historia y a introducirnos en ella de manera como si se estuviera viviendo. En un ARG los participantes tienen que encontrar pistas y hacer esfuerzos para resolver diversas pruebas. Estas pistas se pueden localizar en diferentes lugares, como en bibliotecas, colegios, edificios, parques, etc. [6] [12] [31]

El origen de este tipo de juegos es dudoso. Se dice que uno de los posibles orígenes fue en 1905 con la historia “The Tremendous Adventures of Major Brown” de G.K Chesterton, la cual parece predecir el concepto de ARG, en el que El Padre Brown, el protagonista de la historia, es un sacerdote católico cuya agudeza psicológica le convierte en un gran detective que es capaz de resolver la mayoría de los casos de forma racional.

En la novela “The Magus” en 1965, John Fowles sí que refleja el género que hoy en día conocemos por ARG. Cuenta la historia de Nocholas Urfe, que enseña en un colegio de una isla griega y se ve envuelto de ilusiones psicológicas de un mago. “The Magus” demuestra que la vida son infinitos caminos y que siempre se debe elegir el que más te apetezca andar. Esta novela ayudó a promocionar el interés, en los años sesenta, por el psicoanálisis y la filosofía mística.

Otro antecesor posiblemente haya sido el tipo de libro de “Elige tu propia aventura”. Son series de libros infantiles y juveniles muy populares durante los años ochenta y noventa. Es un tipo de libro en el se tiene que elegir entre diversas opciones que ramifican la trama y conducen, a su vez, a nuevas opciones que desembocaban en distintos desenlaces. Son libros muy educativos, ya que estimulan las ganas de leer y la capacidad de decisión, de esta manera los niños comprenden que sus decisiones pueden ser relevantes. Han sido traducidos a 38 idiomas distintos. Estos libros se han usado en clases desde la escuela primaria hasta la universidad.

Los ARGs son en realidad una búsqueda, desde el punto de vista del jugador. Hay que seguir una serie de pistas, desde unos rompecabezas sencillos a otros no tanto, de una determinada historia y llegar a encontrar lo que se busca.

Los ARG utilizan el mundo real como soporte. La historia y la información no tienen por qué transcurrir de forma lineal, lo que en otro tipo de

juegos es una obligación, y por otro lado, la información puede estar divida en fragmentos repartidos por varios lugares, es decir, por ejemplo, lo que alguien encontró en París puede tener transcendencia en San Francisco. La filosofía de este tipo de juegos es ir resolviendo un gran rompecabezas y/o caso en el que el jugador llega a confundir la historia con la realidad, para lo cual, se emplean todos los medios posibles existentes en el mundo real, como páginas web, correos electrónicos, SMS, actores o cualquier medio que comunicación con el que se pueda llegar hasta el jugador. Y no solo eso, sino que además los jugadores pueden comunicarse en tiempo real, con otras personas que también tienen acceso a esa información, sin importar la localización geográfica. Es una característica muy importante el poder de "La mente colmena". La tecnología actual permite que los jugadores puedan comunicarse, mediante foros, por ejemplo, para que puedan investigar la historia de forma conjunta, enlazar todas las pistas y averiguar el final.

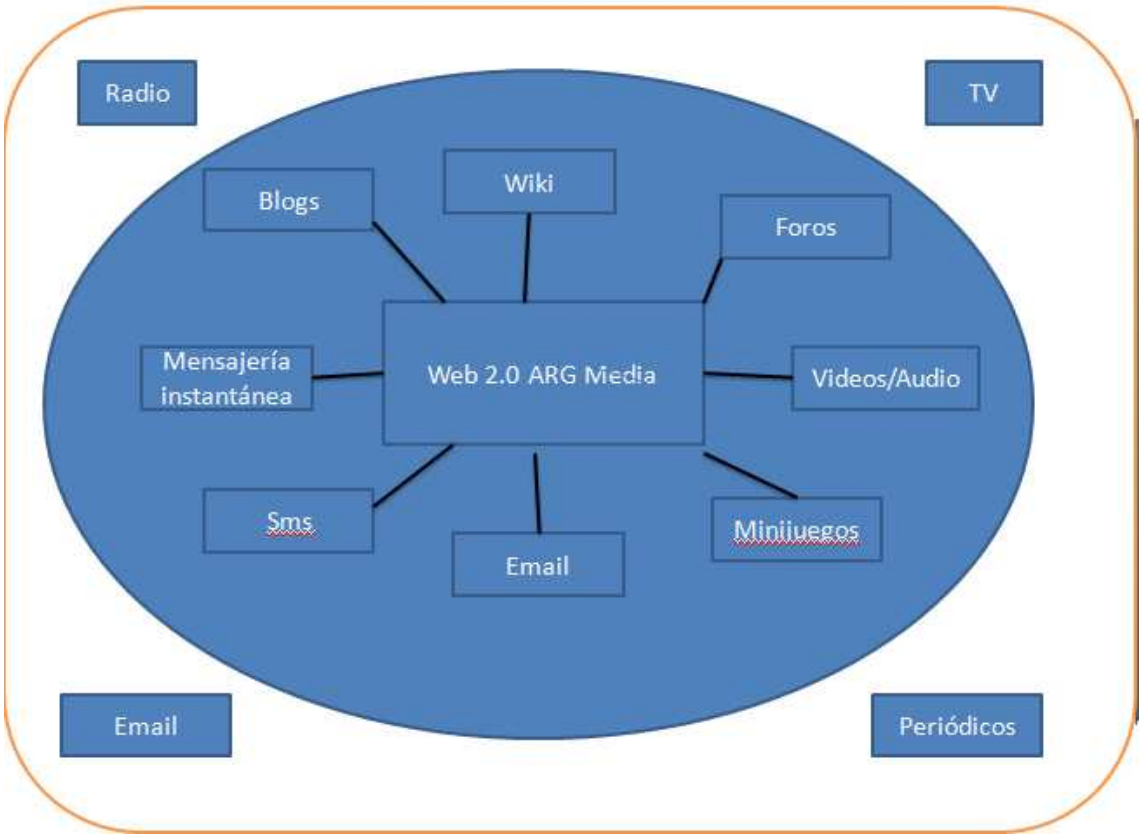

Ilustración 5 Web 2.0 en el diseño y desarrollo de ARGs

El objetivo es que el jugador, si quiere, pueda estar las 24 horas inmerso en el juego.

Para que a un juego se le considere de realidad alternativa el tiempo verbal de la acción debe ser el presente, ya que los jugadores presencian la historia en directo. El jugador se llega a convertir en el protagonista de la

historia, que puede verse modificada en cualquier momento. El juego no reconoce que es un juego.

La historia presenta un mundo completo, números de teléfono, direcciones de correo electrónico, páginas web, etc. Como hemos dicho anteriormente, el juego se realiza en tiempo real y no se puede jugar de nuevo. También existen ARGs que no son multijugador, y quien juega solo se relaciona con personajes que pertenecen al juego cuando envían sus mails o se visitan sus webs, en los cuales sí se podría volver a jugar.

Pueden encontrarse muchas similitudes entre los Juegos de Realidad Alternativa, los Juegos de Rol y las campañas publicitarias, ya que muchas compañías buscan hacer sus productos conocidos a través de un ARG, tales como el grupo de rock "Nine Inch Nails", el automóvil Audi A3, el motor de búsqueda MSN Search, el videojuego GUN, la consola Xbox 360, la película "Pirates of the Caribbean: Dead Man's Chest" y el sistema operativo Windows Vista.

Este tipo de juegos está adquiriendo mucha fama, además de por los motivos anteriormente descritos, porque aparta de la vida rutinaria a ciertas personas, al hacer cruzar los límites entre el mundo real y el virtual, consiguiendo que se vivan nuevas situaciones que en la vida real, quizá, no se podrían dar, como por ejemplo, salvar el planeta.

Jane McGonigal [13] [14] es una *puppermaster*, diseñadora e investigadora de videojuegos. Estudia las carencias y necesidades del mundo actual y trabaja para arreglarlas. Su lema es "*Reality is broken, game designers can fix it*". Piensa en cómo los juegos pueden cambiar la vida de las personas y cómo pueden hacer un mundo mejor. Realiza los siguientes cálculos: en el planeta se pasan 3 billones de hora a la semana jugando al ordenador y a videojuegos, y al ser esto mucho tiempo, es el suficiente para cambiar la vida de las personas, incluyendo la posibilidad de que se pueda salvar el mundo real. A lo largo de su carrera ha dado conferencias explicando cómo los juegos pueden cambiar a las personas, al mundo, e inclusive, hacer que mejore. En ellas pedía colaboración a ciudadanos normales con ideas para ponerlas en común y lograr dichos objetivos. Después de la conferencia TED [15] en febrero de 2010, recibió entre 50 y 100 emails a la semana de gente interesada en conocer el impacto de este tipo de juegos y en participar y crear nuevos.

Los ARGs suelen ser juegos multijugador, los cuales solo se pueden jugar una vez y desde distintas partes del mundo. Algunos ejemplos representativos son:

a) Perplex City

Mind Candy presentó un juego de realidad alternativa en el que los jugadores tuvieron que buscar "El Cubo de Receda", un artefacto muy

importante para los ciudadanos de Perplex City, ciudad extraterrestre donde fue robado. El Cubo había sido enterrado en algún lugar de la Tierra al que los ciudadanos de esta ciudad les era imposible llegar, por eso reclamaron la ayuda de ciudadanos de la Tierra para encontrar el objeto tan importante. A la persona que lo encontró, se le premió con 150.000€.

La historia fue contada a través de blogs, estando repleta de rompecabezas que había que resolver, algunos de mucha duración, lo que hizo que el juego durara casi dos años, desde abril de 2005 hasta febrero de 2007. [16]

b) The Lost Ring

Fue un juego desarrollado en marzo de 2008, cuyo *puppetmaster* fue Jane McGonigal, que unió a jugadores de todo el mundo para descubrir los secretos de Olimpia. La historia se centró en Adriadne, el último atleta olímpico de un universo paralelo. Un día Adriadne se despertó y se dio cuenta de que no recordaba nada. En su brazo tenía un tatuaje en esperanto que decía: “Encuentra el último anillo”. Los jugadores pronto se dieron cuenta de que había atletas por todo el mundo.

Se compartió todo tipo de pistas para resolver el misterio de los atletas, se usaron blogs, webs, Messenger, Google Maps y todo tipo de comunicación. El objetivo era encontrar un anillo en cada continente y con ello se salvaban todos los universos ya que ellos tenían la clave para alinear nuestro mundo con el resto. El 24 de agosto de 2008, los jugadores resolvieron la gran prueba en los Juegos Olímpicos de Pekín y en cada uno de los continentes, con ello, lograron salvar el mundo de la destrucción. [17]

c) Top secret dance off

En este ARG lo primero que piden es crear un perfil para identificarse como bailarín del juego.

Ilustración 6 Jugador de "Top secret dance off"

"Top Secret Dance Off" es una aventura a la que se puede jugar desde cualquier parte del mundo. No es necesario que se sepa bailar, solo es necesario:

- ✓ Música
- ✓ Una cámara que grabe de forma digital para poder subir vídeos a la web del juego y así compartir tus misiones con el resto de jugadores.

El objetivo del juego es ir completando misiones bailando, incluso hay veces que es necesario realizar los vídeos en casa, otras en la calle o en cualquier lugar imaginable. Se puede jugar solo o hasta con cuatro amigos, pero siempre hay que ir enmascarado. [18]

d) CryptoZoo

En este juego se ha creado un mundo secreto de criaturas extrañas que se mueven muy rápidamente.

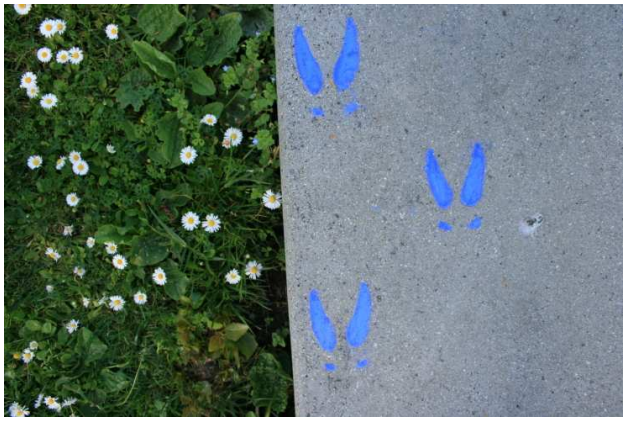

Ilustración 7 Huellas de las extrañas criaturas

La cryptozoografa Natalie Cartwright había recopilado pruebas, fotografías, notas y dibujos de “cryptis” en el mundo salvaje. La ciudad estaba tomada por extrañas criaturas. Había que aprender de ellas.

Se hicieron grupos en ciudades muy alejadas, por ejemplo Londres y Wellington, cuyo objetivo era encontrar los caminos de los “cryptis” y nuevas especies gracias al mapa de CrytoZoo.

Ilustración 8 Jugadores de CrytoZoo

El juego tenía una web donde los jugadores podían escribir en foros, conocer las reglas del juego, ver vídeos, fotos y al resto de miembros que jugaban en todas las ciudades. [19]

e) Superstruct

Jane McGonigal fue la *puppermaster* de este juego. Comenzó el 6 de octubre de 2008, e invitaba a los jugadores a imaginar la vida en el año 2019 y a predecir, tanto ellos como sus familias y amigos, cómo podrían actuar ante una catástrofe en la población.

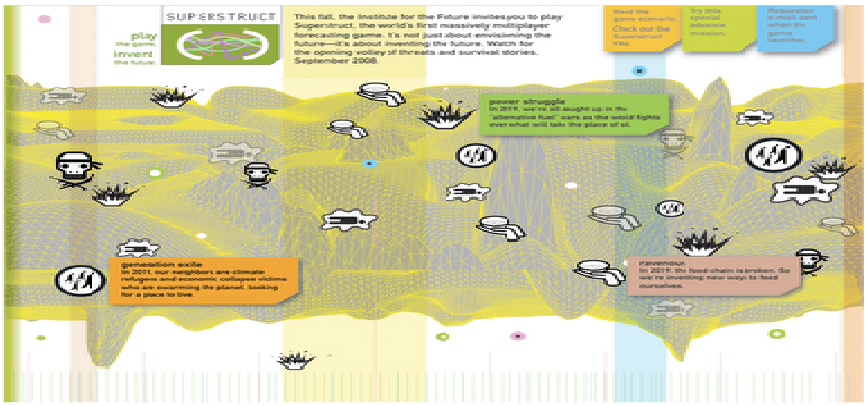

Ilustración 9 Superstruct

Este ARG comenzó con las conclusiones de un superordenador de ficción que, después de un análisis de un año, predijo la extinción de la población humana en 23 años. Una combinación de amenazas en el área ambiental, económica y social.

El juego mostró como sería el mundo en el año 2019, enseñó lo que es vivir en esa fecha y había que buscar la manera de hacer de posible la existencia de los seres vivos. Se trataba de hacer el futuro, un conjunto de superestructuras, inventando nuevas formas de organizar la raza humana y aumentar nuestro potencial humano colectivo.

A “Superstruct” podía jugar cualquier persona, de hecho, cuantas más personas, más ideas se aportaban y mejores predicciones se realizarían. Como ARG, se compartió información en foros, blogs, videos y web. [20]

2.3.2. ARGs pertenecientes a una campaña publicitaria

Consiste en introducir pistas y piezas de rompecabezas en la publicidad de un producto, pasando desapercibidas para las personas externas al juego y siendo algo a estudiar para los participantes. Comenzaron como una nueva forma de hacer conocer un producto. Principalmente estaban destinados a hombres entre 18 y 35 años. No es un juego de un solo jugador, ya que si fuera así, la campaña publicitaria no tendría sentido, pero en este caso tampoco colaboran entre ellos, se juega de forma independiente [6]. Algunos ejemplos son:

a) ¿Dónde está Yago?

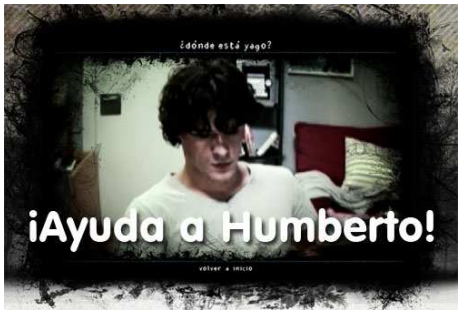

Ilustración 10 Presentación del juego ¿Dónde está Yago?

El juego basado en la serie de televisión "El internado" tiene como argumento la desaparición de Yago, un personaje que estudió en el colegio llamado Laguna Negra. Un amigo suyo intentará encontrarlo con nuestra ayuda mediante juegos y acertijos, además, algunas de las respuestas a los enigmas estaban dentro de los episodios y solo viendo el capítulo y fijándose en todos los detalles se podría pasar a la siguiente fase. [21]

b) Lost

Cuenta con una trama paralela a la serie "Perdidos". Sam Thomas es el protagonista, busca desesperadamente a su novia Sonya, azafata del vuelo 815. Oceanic comunica la cancelación de la búsqueda del avión ese mismo día en su web. Sam introduce un mensaje en la web de Oceanic, en el que nos mostrará una fotografía de Sonya, donde encontraremos las primeras pistas del juego. Según vayamos avanzando, podremos ver más videos.

Podremos entrar en la página web de la compañía del avión siniestrado e incluso reservar asiento y mediante Google Maps seremos capaces de situar la isla donde cayó el avión.

Este juego se dio a conocer cuando no estaban emitiendo Perdidos, así se mantenía a los fans pendientes de que empezase la siguiente temporada [22].

c) Audi A3

Nisha, Ian, Virgil y Arclight formaban un equipo de atracadores y protagonizaron el mayor robo de obras de arte del mundo. Este juego que sirvió para dar publicidad al nuevo A3. Durante el juego, se pudo ver una cámara de seguridad grabando una sombra, la cual estaba escondiendo tarjetas SD con información dentro de seis Audis A3.

El juego consistió en que jugadores de la vida real, denominados *retrievers*, deberían encontrar dicha información.

Si se llegaba a juntar toda la información contenida en los coches, se podía llevar a cabo el gran atraco a las obras de arte. [23]

d) The Beast

Es un juego tipo ARG creado para promocionar, la película de Steven Spielberg, "Inteligencia Artificial". Durante doce semanas, en el año 2001, se convirtió en uno de los primeros ARGs con más influencia. Las pistas que se incluyeron daban acceso a una historia que ocurriría en 50 años. Todos los rompecabezas estaban hechos para crear la ilusión de que la historia ocurriría. [24]

e) Batman, El Caballero Oscuro

Para la promoción de la nueva película de Batman, los fans de la película tenían que encontrar móviles, una carta del Joker e instrucciones que decían lo siguiente que había que hacer en varias ciudades de los EE.UU.

Como puede verse en la siguiente imagen las instrucciones estaban escondidas en sitios tan extraños e improbables como una tarta.

Ilustración 11 Tartas donde estaba oculta información del ARG

En un cierto momento, la persona que encontró uno de los móviles recibió una llamada pudiéndose escuchar lo siguiente:

"¡Buen trabajo, payaso! Mantenga este teléfono cargado y con usted siempre. No me llame. Le llamaré…. "

Con estas palabras el Joker, un personaje de ficción, se convertía en real [25].

f) Suceso Tierra

Es el primer juego de realidad alternativa en España. En él, Adriana Garrido, periodista, está investigando la desaparición de Lucas Montano,

que había desparecido cuando estaba intentando descubrir la localización donde habían caído unos meteoritos en extrañas circunstancias.

Adriana nos pide nuestra ayuda para desvelar el lugar donde está Lucas.

En YouTube podemos ver un video de su secuestro.

Ilustración 12 Video del secuestro de Lucas

Este juego fue la campaña que desarrolló la firma Toyota para promocionar su nuevo modelo Auris. [26] [27]

g) I love bees

Es un juego multijugador y multiplataforma. Sirvió como publicidad subliminal ya que introducía a los jugadores en el mundo de "Halo2" mediante un *trailer*. El juego usaba cabinas telefónicas para enviar la información que necesitaban las personas. Durante la campaña los jugadores debían responder a las preguntas hechas en distintas cabinas en varios países del mundo. Cada semana se difundía un nuevo capítulo, el cual, era esperado por los jugadores para poder seguir en la investigación. Fue seguido por diez millones de jugadores que participaban en misiones, comunicándose entre ellos para resolver el misterio. "Halo2" se convirtió en un fenómeno, atrajo y mantuvo la atención de los jugadores gracias a que los propios jugadores sentían que era más que un juego. Vendió 125.000 copias el día del lanzamiento y, en total, vendió 7 millones de copias.

La campaña "I love bees" ganó en 2005 el premio "Game developers" por la innovación causada. [28]

Ilustración 13 Juego Halo 2

2.3.3. ARGs serios

Son juegos con las mismas características que los anteriores, pero que ponen a los jugadores ante posibles problemas del mundo real, como el aumento del agujero de la capa de ozono o el agotamiento del petróleo en el mundo.

El objetivo de este tipo de juegos es que los jugadores puedan buscar nuevas ideas para sobrevivir ante la nueva situación.

a) World Without Oil

Se plantea el caso de escasez de petróleo en el mundo. La página del juego se divide en secciones. Hay una para estudiantes donde se pueden ver lecciones con instrucciones para que puedan estudiar la situación a la que se enfrentan. También hay directrices para los profesores para que trabajen con sus alumnos y planteen cuestiones acerca de la energía en la sociedad, su uso y la sostenibilidad. Para ello, se usan blogs, videos, mails y buzones de voz. [29]

b) Evoke

Juego organizado por Jane McGonigal en 2010, es un ARG cuyo objetivo es hacer que los jóvenes del mundo, especialmente los de África, aporten soluciones para los problemas reales que existen en el mundo, como el hambre, desastres naturales, enfermedades, etc. Durante diez semanas los jugadores viajaron alrededor del mundo con el objetivo de que conocieran los problemas y buscaran una solución. El primer capítulo simuló la mayor hambruna en Tokio, que duraría diez años.

El juego consistía en completar 10 retos, y aquellos que lo lograran obtendrían un certificador del Banco Mundial, Instituto Social de la Innovación. A los mejores se les daría la oportunidad de recibir becas para compartir su visión [30].

2.3.4. ARGS educativos

Como hemos dicho anteriormente, el juego es una nueva forma de enseñanza. La idea de usar los juegos como método educativo ha adquirido más fuerza recientemente. [33]

Aunque existen pocos estudios sobre la efectividad de los juegos de ordenador en la educación y también pocos que los comparan con otras formas de enseñanza, como clases magistrales, tutoriales o laboratorios, los juegos de realidad alternativa pueden ser muy útiles en un contexto educativo y hacer que los estudiantes puedan explorar nuevas ideas y puntos de vista.

A pesar de que la principal aplicación de los ARGs se encuentra en el mundo del marketing y de la publicidad, se comienzan a usar en el mundo de la enseñanza porque: [7]

- ✓ Existe la posibilidad de resolución de problemas a varios niveles: el estudiante decide por qué nivel empezar.
- ✓ Existe la relación progreso-recompensa: puede ser usado en una evaluación.
- ✓ Usan dispositivos que son capaces de narrar una historia: muy útil en asignaturas de historia o noticias.
- ✓ Se regula la entrega de nuevos problemas: de esta forma se mantiene al jugador atraído por el juego.
- ✓ Activan y potencian la comunicación entre los miembros del juego.

Algunas de las características por las cuales los ARGs se usan como forma de enseñanza son:

Los ARGs son inmersivos, es decir, si la historia está bien escrita tienen la característica de introducir al jugador en el juego de forma envolvente. Requiere su colaboración y resolución de problemas.

- ✓ Los ARGs pueden ser usados para ayudar a la gente a construir un **prerrequisito** de conocimiento necesario para el siguiente nivel de aprendizaje. Es decir, preparan al jugador intelectualmente para el aprendizaje de un tema posterior.
- ✓ Los ARGs pueden ayudar a hacer **menos áridos** ciertos temas, como por ejemplo, temas legales. Si se comparte la información a través de una historia y con un formato de juego, como un sistema de puntuación o como una competición.
- ✓ Los ARGs pueden facilitar un aprendizaje **más rápido** y hacer que el conocimiento adquirido se retenga **más tiempo** que con sistemas tradicionales.
- ✓ Los ARGs no necesitan demasiados recursos tecnológicos para crear un buen juego, tan solo es necesario una buena historia.

Algunos ejemplos de ARGs educativos son:

a) eMapps Proyect ("Motivating Active Participation Of Primary Schoolchildren in Digital online Tecnologies for Creative Opportunities through Multimedia")

Este juego combinó juegos online y tecnologías móviles para demostrar que un ambiente propicio para el aprendizaje puede ayudar a niños, en este caso entre 9 y 12 años, de los nuevos países de Europa. Durante el juego había que:

- ✓ Construir una comunidad generando su propia cultura y comunicación con otros países.
- ✓ Usar herramientas interactivas que ayudaran a integrar el uso del ICT (Tecnología de Información y Comunicaciones) en la entrega de currículum.
- ✓ Crear un mapa basado en geografía, historia y herencia. Accesible desde móviles y que pudiera ser continuamente actualizado según jugaran nuevos miembros de los países de Europa.

b) ARGOSI Proyect

Es un juego creado por "Joint information Systems" en Reino Unido. Estaba enfocado hacia la obtención de bibliotecas y habilidades relacionadas con la información. Además estaba orientado hacia la creación de redes sociales. Este proyecto identificó seis elementos que pueden motivar si se juega a un ARG y consigue que los estudiantes quieran seguir inmersos en el juego:

Elemento	Posible implementación.
Comunidad	Actividades de colaboración, herramientas de comunicación.
Competición	Premios.
Completar	Piezas o pistas que necesitan unir.
Creatividad	Desafíos creativos.
Narración	Una historia que contenga un misterio.
Resolución de puzles	Desafíos basados en la resolución de puzles.

Ilustración 14 Elementos que motivan a la hora de jugar a un ARG

c) Torre de Babel

Este juego tenía como objetivo motivar el plurilingüismo a los estudiantes. La historia se basaba en llevar a los estudiantes al futuro para salvar los lenguajes que estaban bajo la amenaza de desaparición. Solo

podían salvarlos si los estudiantes colaboraban entre ellos. Durante este ARG:

- ✓ Se practicaba el lenguaje elegido de forma muy parecida a una situación real.
- ✓ Se permitía a los estudiantes expresarse sin la preocupación de cometer errores.
- ✓ Los estudiantes se esforzaron más ya que el entorno intimidaba menos que en una clase tradicional.
- ✓ Se aprendía a través de tareas que hacían que los estudiantes se inventaran soluciones creativas a problemas dados.

2.4. Aventuras gráficas con usos educativos

La dinámica de este tipo de juego consiste en un personaje que se mueve por un escenario dado y debe ir avanzando por el mismo a través de la resolución de diversos rompecabezas, planteados como situaciones que se suceden en la historia, interactuando con personajes y objetos a través de un menú de acciones.

El concepto clásico de aventura gráfica incluía la visión de los personajes en tercera persona la mayor parte del juego, aunque algunas de las aventuras gráficas más importantes se planteasen en primera persona.

Los antecesores de las aventuras gráficas son las aventuras conversacionales, juegos que contaban una historia interactiva a través del texto en la pantalla, sin ningún tipo de imágenes. Consistían en que el juego describía una situación-problema, se tecleaba la respuesta-orden y si la respuesta era correcta, por ejemplo, si la orden correcta era abrir ventana, la historia proseguía, sino no avanzaba hasta que no se diera la orden correcta. Las más conocidas fueron las de la saga “Zork”: “Zork I”, “Zork II”, “Zork III”, “Beyond Zork”, “Zork Zero”, creados por Infocom. Se caracterizó por ser un juego con gran calidad de la historia como analizador sintáctico de texto, ya que las órdenes no se limitaban a simples verbos-nombre, por ejemplo, “golpear mazo”, sino que añadía la correcta preposición o conjunción, “golpear el saco con el mazo”. [32]

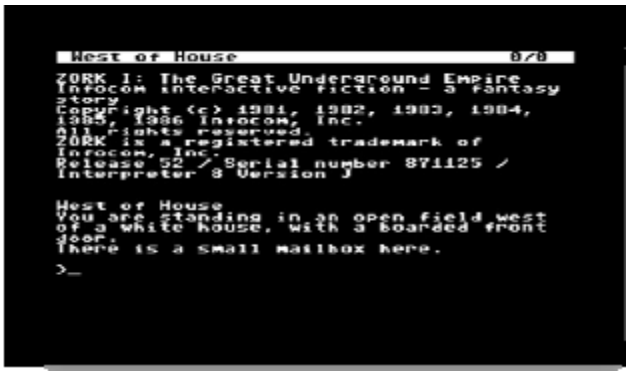

Ilustración 15 Aventura conversacional

A medida que la tecnología progresaba, empezaron a aparecer aventuras de texto con gráficos, como son, por ejemplo, “Mistery House” de 1980, una intriga policiaca inspirada en la novela de Agatha Christie “Diez Negritos”. Se la consideró el primer juego de aventura con gráficos.

En 1984 aparece un juego denominado “King's Quest”, en el que las órdenes se tecleaban pero por primera vez se podía interactuar con los gráficos. Además se introdujo la novedad de la tercera persona, apareciendo un personaje animado que el jugador manejaba.

En 1987 LucasArts creó aventuras gráficas con el motor SCUMM, “Script Utility for Maniac Mansion”. Se pasó de usar el *text-parser* a usar el *point&click*. Creó la aventura gráfica “Maniac Mansion” en la que en la parte inferior de la pantalla aparecerían una serie de verbos de acción que solo habría que activar con un *click* de ratón, lo que causó un gran éxito en este tipo de juegos, y al existir un límite a la hora de elegir las acciones, se produjo una gran simplificación a la hora de jugar.

La década de los noventa, fue la etapa de oro de este tipo de juegos.

2.4.1. Características de una aventura gráfica

Las aventuras gráficas se caracterizan por:

a) Mensaje

El mensaje en una aventura gráfica es el texto que aparece en la pantalla que se corresponde con los diálogos de los personajes.

b) Acciones

En el transcurso del juego el protagonista puede realizar distintas acciones. Entre ellas se encuentran las siguientes:

- **Mirar/Observar/Examinar:** El protagonista de la aventura podrá examinar personajes y objetos.
- **Coger/Abrir/Cerrar/Usar:** El protagonista de la aventura podrá recoger un objeto para incorporarlo al inventario, abrir y cerrar puertas o usar los objetos con personajes o incluso otros objetos.
- **Hablar:** El protagonista de la aventura dialogará con el personaje elegido. Gracias a los diálogos obtendrá información que será necesaria para varias situaciones en las que se encontrará.

c) El inventario

El inventario está formado por todos los objetos que recoges a lo largo del juego y de los que puedes servirte para avanzar en la aventura. En

cualquier momento se puede consultar los elementos que se tiene a disposición en el inventario.

d) Sistema de pistas

En ocasiones estos juegos llegan a desanimar ya que a veces es difícil encontrar la solución correcta. Es por ello que se incorpora a este tipo de juego un sistema de pistas.

Además, todas o prácticamente todas, se caracterizan por tener un mapa del escenario en el que se mueve el personaje protagonista.

2.4.2. Ventajas y desventajas de las aventuras conversacionales frente a las gráficas

Ventajas:

- ✓ Son mucho más fáciles de programar que las aventuras gráficas.
- ✓ Si cuentan con un buen argumento y guión, logran crear más ambiente.
- ✓ Pueden transmitir mucho mejor las emociones y dar rienda suelta a la imaginación del jugador.
- ✓ Las aventuras gráficas no desarrollan tanto la imaginación del jugador como las aventuras conversacionales.

Desventajas:

- ✓ Hay mucha gente a la que no le gusta leer.
- ✓ Un mal guión puede hacer que el jugador se desinterese por el juego.
- ✓ Es necesaria una revisión concienzuda de la ortografía.
- ✓ Si son gráficas, atraen más al jugador y la interacción es más directa, gracias a que se pueden visualizar los escenarios.

2.4.3. Aventuras gráficas educativas de la compañía CMY

Las personas que forman esta compañía han realizado una serie de aventuras gráficas infantiles después de diversos estudios. Profesores del Eurocolegio Casvi y diseñadores gráficos han trabajado juntos con el fin de realizar juegos educativos que ayuden a los niños a repasar la materia impartida en el curso anterior. Vieron que los niños dedicaban mucho tiempo a jugar con video-consolas y vieron la facilidad que tenían para el uso de ordenadores y la motivación inicial que suponía para ellos el proponerles hacer algo delante de un ordenador, así que se plantearon aprovechar esa motivación para su aprendizaje [34].

Dichas aventuras se centran en un personaje protagonista, el cual es el encargado de acompañar al niño a lo largo del juego, adivinando poderes de piedras mágicas, luchando contra malvados piratas, viajando en el tiempo, etc. El niño debe superar los obstáculos mientras, que sin darse cuenta, reforzará los conocimientos aprendidos, basándose en valores de igualdad, respeto por los demás, la protección del medio ambiente y, por encima de todo, en la no violencia. Todas las aventuras de este estilo tienen la misma estructura: un pequeño vídeo en el que se introduce el suceso ocurrido y cuál va a ser la misión a realizar. Posteriormente sale el menú principal, donde se puede elegir la opción que más interese.

a) Aymun y los piratas mechones

Es un programa dirigido a niños cuyo fin es enseñar cálculo mental, composición y descomposición de números, operaciones con decimales, ortografía, etc. Como toda aventura, la metodología consiste en avanzar en el juego para conseguir los objetivos, a través de la resolución de las actividades educativas integradas a lo largo de la aventura. Para ello, cuenta con un sistema de ayuda inteligente, que aporta las pistas necesarias para que el niño pueda avanzar siempre. Se caracteriza por tener una interfaz dinámica, secuencias cinematográficas en 3D, personajes que hablan, subtítulos en castellano y una banda sonora, lo que hace atractivo el juego al niño.

Ilustración 16 Pantalla del juego Aymun

b) La aventura de Don Quijote

Este juego basado en las aventuras de El Quijote, permitirá a los niños conocer la novela de Cervantes, motivándoles para su posterior lectura.

Para ayudar al famoso personaje habrá que ir resolviendo una serie de actividades. Con la ayuda de Cervantes el niño obtendrá las pistas necesarias para proseguir con la historia. Todos los gráficos están hechos en 3D.

Ilustración 17 Don Quijote de la Mancha

c) Nico en el planeta Robot

En esta aventura hay que viajar al futuro junto con Nico y su perro, Peludo, y ayudarles a salvar al planeta Tierra. En sus distintas etapas se integran actividades didácticas que el niño deberá superar para no perder la dinámica del juego. Estas actividades tienen que ver con ortografía, resolución de problemas matemáticos o compresión oral y escrita.

Ilustración 18 Nico en el Planeta Robot

2.5. ¿Qué es un algoritmo?

En la naturaleza existen muchos procesos que pueden ser considerados algoritmos, incluso hay veces que no nos percatamos de que existen: son por ejemplo la circulación sanguínea, la digestión, los ciclos cósmicos, etc.

También existen muchos algoritmos en la vida cotidiana, por ejemplo, una receta de cocina, ya que para poder preparar dicha receta, hay que seguir una serie de instrucciones en un cierto orden.

Se entiende por algoritmo una lista bien definida, ordenada y finita de operaciones que permite hallar la solución a un problema.

El lenguaje algorítmico es aquél por medio del cual se realiza un análisis previo del problema a resolver y que está orientado a encontrar un método que permita resolverlo [35].

2.5.1. Origen de los algoritmos

El origen de la palabra algoritmo viene del nombre en latín del matemático árabe "Abu Ja'far Muhammad ibn Musa al-Khwarizmi", el cual escribió sobre los años 800 y 825 su obra "Quitad Al Mugabala" donde se recogía el sistema de numeración hindú y el concepto del cero. Aunque fue Fibonacci quien tradujo la obra al latín y la inició con la palabra Algoritmi Dicit [1].

A lo largo de la historia, muchos matemáticos han inventado algoritmos que hoy en día son muy conocidos y se siguen utilizando ya que son la mejor manera para resolver el problema en cuestión, como, por ejemplo, el "algoritmo de Euclides" para calcular el máximo común divisor de dos enteros positivos o el "algoritmo de Dijkstra" para encontrar el camino más corto entre un origen y un destino.

2.5.2. Características de los algoritmos

Las dos características que deben cumplir todos los algoritmos son:

- ✓ Un algoritmo debe ser preciso e indicar el orden de realización de cada paso.
- ✓ Un algoritmo debe estar definido. Si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez.

En el mundo de la programación los algoritmos son independientes del lenguaje usado a la hora de escribir el código necesario. También son independientes de la máquina que ejecuta dicho algoritmo. Esto es muy importante destacarlo ya que hay que recalcar que un mismo algoritmo puede ser implementado en cualquier lenguaje de programación.

Un algoritmo es mejor cuantos menos recursos consuma, más fácil sea de programar, si es corto, fácil de entender, robusto, etc.

2.6. Experimentos previos

Un campo muy importante, e incluso podría decirse que esencial, de la programación es el diseño de algoritmos y la resolución de problemas mediante éstos, que normalmente, los estudiantes deben aprender solos, siendo una tarea difícil de realizar.

Según estudios de la universidad de Villanova [8], en Estados Unidos, la realización de ciertos tipos de juegos, como por ejemplo de ingenio y lógica, ayudan a aprender a pensar siguiendo una serie de pasos ordenados para resolver un gran problema. Los juegos muestran que el diseño de estrategias algorítmicas pueden ayudar a adquirir herramientas, habilidades y a desarrollar la creatividad para resolver problemas en el área de la programación. Además de ser didácticos, son entretenidos e interesantes, hacen que el jugador, en este caso, un estudiante, se esfuerce en pensar de forma algorítmica.

Todo problema algorítmico es un reto para su diseñador. Algunos resultan inmediatos y fáciles de resolver, otros son bastante complejos y poco intuitivos, por lo que muchas veces resulta más adecuado y cómodo utilizar algún algoritmo ya diseñado que empezar desde cero. Diseñar un algoritmo es una tarea bastante creativa donde se necesitan ciertos conocimientos y donde la experiencia del diseñador juega un papel fundamental.

La investigación en esta área ha permitido descubrir un conjunto de métodos de diseño hacia los cuales puede orientarse la realización de muchos algoritmos.

Como hemos dicho anteriormente, el diseño de un algoritmo que resuelva un problema es una labor difícil y complicada. Una forma de facilitar esta tarea consiste en recurrir a técnicas conocidas de diseño de algoritmos, es decir, a esquemas muy generales que pueden adaptarse a un problema particular al detallar las partes generales del esquema.

Vemos a continuación algunas técnicas algorítmicas:

2.7. Técnicas algorítmicas

Si el programador, o el aspirante a programador, tiene un recetario de algoritmos de donde puede seleccionar el más adecuado para cada problema, su tarea se ve simplificada.

Algunas de las técnicas de diseño de algoritmos y algunos juegos de ingenio que se resuelven gracias a ellas son:

a) Fuerza bruta

Consiste en la resolución de un problema sin seguir un orden dado, probando todas las posibles combinaciones. El procedimiento de resolución por fuerza bruta es uno de los menos adecuados para resolver un problema, dada su ineficiencia, pero en ciertas ocasiones, son los únicos que encuentran la solución al problema. Son soluciones directas, pero poco reflexionadas.

Un ejercicio que se resuelve por fuerza bruta es: "Unir los nueve puntos sin levantar "el bolígrafo" del papel usando solo cuatro líneas rectas".

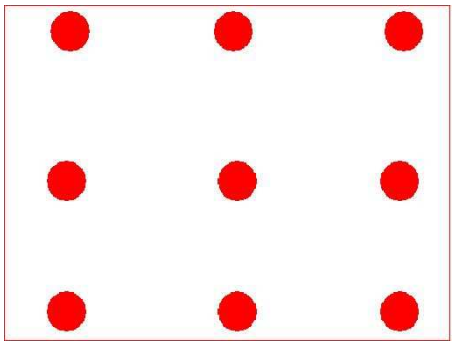

Ilustración 19 Unir puntos

Para poder resolver este acertijo hay que cumplir los requisitos impuestos, y no cumplir los requisitos impuestos por la propia persona que está resolviendo el juego, ya que, por error, se tiende a pensar que no se puede salir del cuadro formado por los puntos.

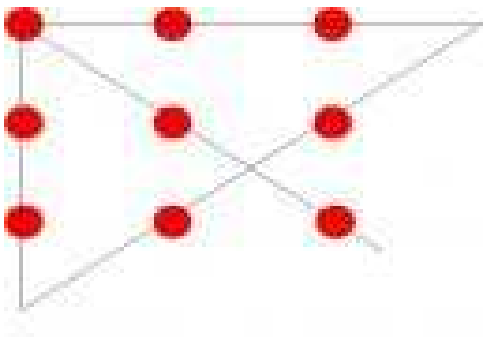

Ilustración 20 Resolución juego unión todos los puntos

Un algoritmo de tipo fuerza bruta es *Selection Sort*. Es un método de ordenación que consiste en buscar el elemento más pequeño de una lista e intercambiarlo con el elemento que está en la primera posición. Luego se volvería a buscar el elemento menor y se intercambiaría con el segundo elemento, y así hasta que la lista estuviera ordenada.

En pseudocódigo este algoritmo sería:

```
for i=1:n-1
   minimo = i;
   for j=i+1:n
      if lista[j] < lista[minimo]
        then minimo = j
      end if
   end for
   intercambiar(lista[i], lista[minimo])
end for
```

Como se puede apreciar, no es un algoritmo eficiente: sabemos que la lista acabará ordenada, pero no con el menor número de pasos. Es apropiado para pequeñas cantidades de datos.

b) Divide y vencerás.

Consiste en dividir un problema en sub-problemas, de la misma clase que el problema general y a continuación dividir estos sub-problemas en otros de nivel más bajo, hasta que pueda llegarse a una solución, luego habrá que combinar las soluciones y obtener la solución del problema general.

Algunas técnicas muy usadas son:

- Quicksort

Es el método más rápido de ordenación, pero es complicado de implementar y es apropiado para cantidades grandes. Tiene una complejidad O(nlogn).[36]

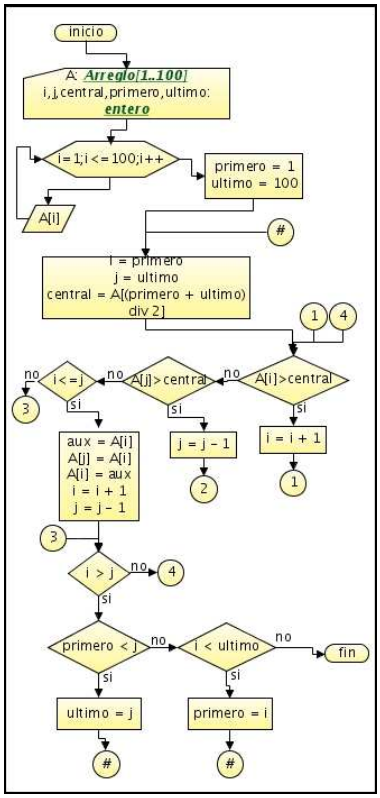

Ilustración 21 Diagrama de flujo de Quicksort

Algunos juegos que usan esta técnica son:

- Triominos: Cuyo objetivo es cubrir un cuadrado de 2*n* por 2*n* con la figura dada. En el juego existe un cuadrado perdido que puede estar en cualquier posición.

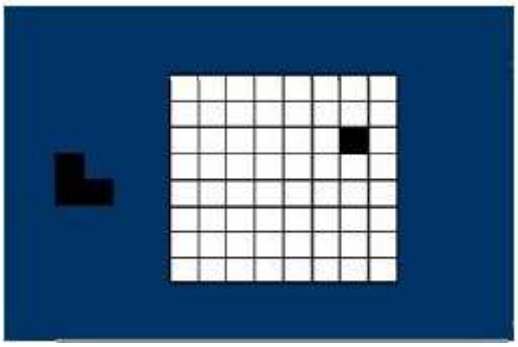

Ilustración 22 Triominio

Las reglas para cubrir el tablero son:

- Caso base 1: Tablero de dimensiones 1x1.

- Caso base 2: Tablero de dimensiones 2x2.

- Dividir el tablero en cuatro bloques iguales. El cuadrado perdido estará en uno de esos cuatro bloques. Suponemos que el cuadrado perdido estará en el cuadrado superior izquierda (por ejemplo) que por inducción sabemos que se puede cubrir y nos centraremos en los otros tres restantes. Subdividiendo los bloques restantes en bloques hasta 2x2, dejando sin cubrir los cuadrados más cercanos al centro de la división en la que el tablero se había dividido en cuatro bloques iguales, quedaría en el centro del tablero la forma de la figura con la que estamos rellenando el tablero, pudiéndose así cubrir con una pieza adicional.

- El problema de las tuercas y los tornillos: Dado un grupo de n tornillos de diferente anchura y n tuercas, el juego consiste en unir las tuercas con el tornillo correspondiente. Al jugador se le permite comparar un tornillo con una tuerca pero no puedes comparar dos tuercas o dos tornillos. El problema consiste en dos vectores. El vector "A" corresponde a los tornillos y el vector "B" a las tuercas. Para emparejarlos usando la técnica de "divide y vencerás" los pasos a seguir son:

1. Elegir un pivote del vector "A"
2. Ordenar el vector "B" a partir del pivote anterior
3. Elegimos el pivote de "B" en base al anterior
4. Ordenamos el vector "A" a partir del pivote de "B"

c) Resta y vencerás.

Inicialmente tenemos un problema difícil de resolver con *n* elementos. La técnica consiste en ir resolviéndolo según se va disminuyendo *n*. Algunos ejemplos:

- Problema de Josephus: Es un problema clásico de este tipo de técnica. Josephus es un judío que los romanos van a matar junto a otros 40. Pero éstos no quieren dar ese placer a los romanos y deciden suicidarse. Se colocan en círculo, y comenzando por el primero, van muriendo uno de cada tres, así el último que quedase se suicidaría. Pero Josephus no quiere morir. El problema consiste en adivinar dónde tiene que situarse para no hacerlo. Para ello hay que probar cada una de las posiciones posibles en el juego. La posición en la que Josephus no moriría sería la 31.

- Descubrir cuál es la moneda falsa con una balanza: Se dan n monedas, siendo una falsa y pesando distinto de las verdaderas. Se debe encontrar la moneda falsa, utilizando una balanza el menor número de veces. Para resolver el juego en primer lugar dividimos todo el grupo en tres montones de igual número de monedas. La primera vez que pesamos se comparan dos montones, si resulta que la balanza está equilibrada la moneda distinta se encuentra en el grupo, si no es así, es decir, los

platillos están desequilibrados, entonces el grupo que más pese contiene la moneda distinta.

- Tres maridos celosos y sus respectivas esposas tienen que cruzar el río en un bote que solo puede llevar a dos personas en cada viaje. ¿Cómo logran cruzar todos ellos el río de forma que nunca una mujer queda en compañía de uno o dos hombres si su marido no está presente? Al aplicar la técnica de resta y vencerás se llega a la solución de la siguiente forma: Se llamara a los tres hombres "A", "B", y "C". Primeramente A y su esposa cruzan el río. "A" vuelve. La esposa de "B" y la esposa de "C" cruzan el río. La esposa de "A" vuelve. "B" y "C" cruzan el río. "B" y su esposa vuelven. "A" y "B" cruzaron el río. La esposa de "C" regresa. Las esposas de "A" y "B" cruzan el río. "C" Regresa. Y finalmente, "C" y su esposa cruzaran el río.

d) Transforma y vencerás.

Necesitaremos hacer una transformación en el problema con el fin de llegar a la solución. Algunos juegos que se resuelven mediante este método son:

- Cifra del César: Consiste en descifrar una o varias palabras. El método consiste en que a partir de una clave numérica, el abecedario se ve modificado. Es decir, si la clave es 4, la letra A pasa a ser la E, la B pasa a ser la F y así con todas las letras.

- El numero enigmático: Dada una secuencia, haciendo una modificación en los números, podemos adivinar el siguiente número en la secuencia, como por ejemplo si tenemos "19, 18, 16, 17, 10, 12, 2…", adivinar el siguiente número según está planteado el problema es complicado, pero si ponemos los números escritos con letras, podremos darnos cuenta de que siguen un orden alfabético. Procediendo de esta forma, se llega a la conclusión de que el siguiente número e la secuencia es el 200.

e) Recursividad.

La recursividad se usa cuando hay que repetir cierto tratamiento o cálculo pero el número de repeticiones es variable.

- Torres de Hanoi: Es un juego típico usado para el aprendizaje de algoritmos. El juego consiste en n discos y tres pilas. Los discos están dispuestos de mayor a menor longitud de radio. El objetivo es llevarlos a la pila B, utilizando la C, de uno en uno y sin colocar un disco mayor sobre uno menor.

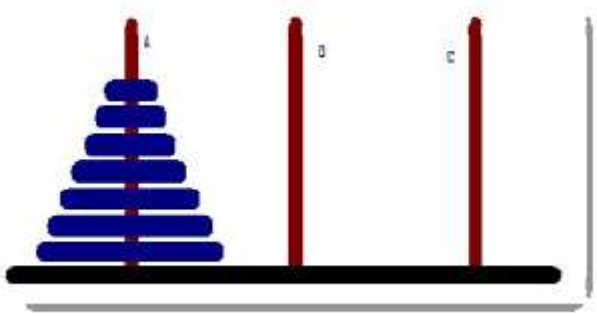

Ilustración 23 Torres de Hanoi

Si numeramos los discos desde 1 hasta n, y llamamos “A” a la primera pila de discos (origen), “C” a la tercera (destino) y “B” a la intermedia (auxiliar) y a la función le llamaríamos *hanoi (origen, auxiliar, destino)*, como parámetros, la función recibiría las pilas de discos. El algoritmo de la función sería el siguiente:

1. Si origen == {0}: mover el disco 1 de pila origen a la pila destino (insertarlo arriba de la pila destino); terminar.
2. Si no: *hanoi({0...n-1},destino, auxiliar)* //mover todas las fichas menos la más grande (n) a la varilla auxiliar
3. mover disco *n* a destino //mover la ficha grande hasta la varilla final
4. *hanoi (auxiliar, origen, destino*) //mover todas las fichas restantes, {0...*n*-1}, encima de la ficha grande (*n*)
5. terminar

- Forma de salir de un laberinto: A pesar de que existe un método eficaz para salir de un laberinto, el de la mano derecha, usando la técnica del *Backtraking* también lograríamos salir. Es una técnica recursiva que consiste en que dada una posición inicial, se observan todas las posibles salidas, una a una, viendo el camino que tiene salida y el que no, volviendo al punto inicial y marcando la opción que conllevaría a una salida, se avanzaría por ella hasta encontrar un nuevo cruce, y el proceso se volvería a repetir hasta que se fuera capaz de salir del laberinto.

- N-reinas: Max Bezzel, ajedrecista alemán, lo propuso en 1848. Conociendo los movimientos que una reina puede realizar sobre un tablero de ajedrez de *n* x *n*, si tenemos N reinas en él, situadas de forma que se den jaque, el problema consiste en colocarlas de forma que no lleguen a darse jaque. Una posible solución al tablero de *n*=8 es:

Ilustración 24 Solución tablero *n*=8 reinas

2.8. Conclusiones

Aprender algo nuevo es complicado la mayoría de las veces. En muchas ocasiones es necesario ser muy insistente para llegar a comprender lo que se estudia y puede resultar aburrido, llegándose incluso a abandonar la tarea de aprendizaje. Es por ello que hoy en día maestros y profesores buscan técnicas nuevas y actualizadas para sus alumnos, para que éstos tengan interés por la materia, que la aprendan divirtiéndose y para que nunca lleguen a olvidarla.

Este trabajo tiene ese fin, aprender divirtiendo. Según los estudios que se han realizado, los juegos son una forma efectiva de enseñanza, en la que los alumnos se ven motivados a seguir en la tarea de aprender. Este trabajo trata de aportar habilidades en la resolución de problemas, que más adelante se necesitarán conocer cuando se vaya a diseñar un algoritmo en un lenguaje de programación concreto. Al ser un tema bastante abstracto para toda persona iniciándose en el mundo de la programación, es bastante útil dar los primeros pasos mediante un juego.

La programación es una actividad que requiere ingenio, y en la cual aparecen constantemente problemas a ser resueltos. Tener la habilidad de resolver problemas es esencial para ser un buen programador. Dos características importantes de toda persona que resuelve este tipo de problemas son la fuerza mental y la creatividad, las cuales, pueden adquirirse con un entrenamiento adecuado. Además, las soluciones de algunos problemas son extensas y complejas siendo necesario expresarlas como algoritmos.

Los juegos de realidad alternativa son juegos que involucran al jugador en la historia, por lo que resulta muy útil utilizar este tipo de juegos en este trabajo, ya que envolverán al jugador en un ambiente de misterio y se verá involucrado en la historia desde el principio. Para aprender la materia a “impartir”, al tener que resolver pruebas lógicas, preseleccionadas concienzudamente, el jugador tendrá que enfrentarse de primera mano a los problemas planteados. Deberá resolverlos si desea continuar avanzando en la investigación mientras que va adquiriendo las habilidades necesarias en la resolución de problemas que se tenía como objetivo desde el principio del trabajo.

3. DESCRIPCIÓN DEL TRABAJO

3.1. Requisitos
3.2. Argumentos y principios en los que se basa el juego
3.3. Planteamiento general
3.4. Pruebas de funcionalidad y usabilidad
3.5. Problemas y decisiones tomadas

3.1. Requisitos

Para poder analizar los requisitos se va a realizar un breve resumen de lo que queremos alcanzar con este trabajo. El objetivo es proporcionar un software que ayude a desarrollar las habilidades necesarias para la resolución de problemas.

Como se ha comentado en capítulos anteriores, existen distintas técnicas algorítmicas que se usan para la resolución problemas. Aunque en este trabajo no se enseñan dichas técnicas, los problemas que se incluyen en este trabajo se resuelven con ellas sin que el jugador se entere.

Se pretende, por tanto, proporcionar al usuario una herramienta mediante la cual pueda aprender a resolver determinados tipos de problemas sin necesidad de saber que está utilizando un algoritmo para su resolución. Así, posteriormente, cuando se involucre en el estudio de la programación, será mucho más sencillo el diseño y programación de algoritmos más complejos.

Conociendo estas premisas, y teniendo en cuenta que el objetivo de este trabajo no deja de ser un juego y que el fin último de todo juego es divertir a la persona que juega con él, nuestra aplicación parte de los siguientes requisitos:

- ✓ Aportar un entorno en el que los usuarios puedan poner en práctica habilidades de resolución de los problemas que fueran apareciendo.
- ✓ Mantener la filosofía de videojuego, atrayente y divertido, mediante una buena historia, unos buenos diálogos y elementos multimedia, como vídeo y sonido de fondo.
- ✓ Utilización de diálogos sencillos y fácilmente comprensibles. Ya que el juego, a pesar de estar destinado a estudiantes, podría jugar cualquier persona, incluidos niños que quieran ir iniciándose en los problemas de lógica.
- ✓ Recrear un nuevo estilo de videojuego, que fusionara dos estilos muy diferentes. Las aventuras gráfico-conversacionales y los juegos de realidad alternativa. Dicho estilo permitiría al usuario involucrarse de tal forma en la historia que se pudiera confundir con la realidad, es decir, hacer que parezca que lo que sucede es real. Para lograr esa atmósfera, se ha realizado un video introductorio, en el que se aprecia como alguien se acerca a una casa en medio de la oscuridad para secuestrar a Lidia. Saúl, el mejor amigo de la chica, pide ayuda al jugador. Mientras que se mueve por todos los lugares buscando pistas, el jugador es el encargado de la parte de inteligencia. Es decir, ayudar a Saúl en la resolución de los problemas encontrados en la misión de rescate.
- ✓ Las imágenes utilizadas durante todo el videojuego son de personas, objetos y escenarios reales. De esta forma será mucho más sencillo para el jugador imaginar lo sucedido en los lugares donde está Saúl. Para mejorar la simulación de la realidad, se ha añadido una funcionalidad de todo ARG, recepción de emails en la cuenta de correo. (Para poder recibir estos mails, el jugador debe estar conectado a Internet).

- ✓ Una interfaz de juego simple, con todas las opciones a la vista, para que el usuario no se distraiga del juego por no saber utilizar la interfaz y se centre en los misterios a resolver.

3.2. Argumentos y principios en los que se basa el juego

El juego comienza con la petición de ayuda por parte de Saúl. Su amiga Lidia ha sido secuestrada mientras celebraba su cumpleaños en un chalet a las afueras de Madrid. El secuestrador dejó como firma la pared pintada de la habitación donde fue secuestrada, una nota con el número 6 y otra diciendo que no se llame a la policía. De esta forma, la aventura comienza en casa de Lidia, donde aún quedan algunos testigos junto con las pruebas de su secuestro. Saúl decide ir a buscar a la chica mientras el jugador le ayuda desde su PC. Para saber por dónde empezar, Saúl pide al jugador que eche un vistazo a la habitación de Lidia buscando alguna pista valiosa.

Ilustración 25 Portada del juego

A lo largo del juego, Saúl y el usuario forman un equipo de investigación. Mientras el amigo de Lidia se mueve por los escenarios a los que las pistas conducen, el usuario espera la información vía mail. A lo largo de la aventura surgen problemas que Saúl no sabe resolver, por lo que pide ayuda al usuario mediante el correo electrónico. Estas pruebas representan pruebas de lógica que han sido preseleccionadas al ser problemas representativos. Los emails que el usuario va recibiendo son pistas para resolver estas pruebas y que a su vez, son un elemento esencial de todo ARG, cuyo fin es involucrar al jugador en la aventura que está sucediendo.

El objetivo es encontrar a la chica que ha desaparecido gracias a las pistas que aporta Saúl, a la investigación y a la creatividad del usuario resolviendo las pruebas de lógica. Para buscar información, Saúl debe hablar con diferentes personajes, coger objetos interesantes y usarlos en ciertas circunstancias para poder conseguir avanzar en la investigación.

El hilo de la historia es lineal, no se puede volver atrás, ya que se cuentan los hechos según le están ocurriendo a Saúl.

La aventura se divide en escenas, que se corresponden con los escenarios que el amigo de Lidia va visitando y que tienen asociadas las pruebas de lógica que el jugador tiene que superar para encontrar a la chica.

3.2.1. Características del videojuego

Para el diseño y creación de este videojuego se han tomado decisiones que con llevaron a tener las siguientes características.

El lenguaje utilizado para los diálogos se ha intentado que fuese lo más sencillo, cercano y a la vez moderno para los usuarios, con el fin de no aburrir al usuario con diálogos serios que no consigan atraer su atención, ya que, aunque el juego está dirigido a jóvenes y adultos, son los jóvenes los principales destinatarios de este juego.

Como hemos mencionado anteriormente, este videojuego surge de la fusión de una aventura gráfico-conversacional y un juego de realidad alternativa. Para realizar tal mezcla de estilos, se consideraron las siguientes decisiones previas de diseño:

1. Interfaz gráfica semejante a la interfaz de una aventura gráfico-conversacional. Para poder fusionar dos estilos de juego era necesario decidir qué función tendría cada uno. El objetivo de usar un ARG es la inmersión del jugador en la historia, haciéndole protagonista y partícipe de ella, como por ejemplo, recibiendo emails. Pero esta parte es abstracta, ya que todo juego necesita una interfaz de juego visible para el jugador. Es por las ventajas anteriormente mencionadas de las aventuras gráfico-conversacionales por las que se eligió seguir este modelo de interfaz. El lenguaje de programación ha sido Java ya que gracias a las múltiples bibliotecas que componen Java podrían cubrirse las necesidades del videojuego, como por ejemplo, las visuales.
2. Simulación de diversas características de las aventuras gráfico-conversacionales: El usuario “maneja” un personaje por diversos escenarios, el cual, deberá hablar con otros personajes de la historia para conseguir información, deberá además coger diferentes objetos y utilizarlos entre sí con el objetivo de conseguir otros nuevos o iniciar el comienzo de un mini-juego. De esta forma el usuario puede buscar por sí mismo las pistas y tiene la información accesible en todo momento.
3. Simulación de diversas características de los juegos de realidad alternativa: Petición de un nombre de usuario y una dirección de correo electrónico, ya que como en todo ARG, se deben poner en uso diferentes formas de comunicación entre los jugadores. Al ser este un videojuego para un solo jugador, no se usarán ni chats, ni foros. En este caso, se recibirán emails del personaje que investiga la desaparición de Lidia, con la información nueva

conseguida y con peticiones de ayuda a la resolución de ciertas pruebas. Con el fin de que todo parezca que es real se han utilizado imágenes reales de personas, de objetos y escenarios.

4. Solo se podrá pasar de escena si se cumplen las condiciones obligatorias para poder hacer el cambio.
5. No se puede volver atrás en la historia. Aunque en las aventuras conversacionales es muy corriente tener un mapa con los lugares que el jugador puede visitar, en este caso, al introducir el estilo de juego de un ARG, y como en dichos juegos se narra una historia, se decidió que lo más apropiado era que los acontecimientos se sucedieran de forma secuencial, como si se estuviera viviendo una historia.
6. Para poder recibir las pistas correspondientes, el usuario debe estar conectado a internet.
7. Los mini-juegos son pruebas lógicas que debe resolver el usuario para poder avanzar en la aventura. Dichas pruebas lógicas plantean problemas y para poder resolverlos, se tienen que seguir una serie de pasos en un cierto orden. En ocasiones el jugador deberá navegar por internet en la búsqueda de la solución al problema planteado, es decir, en la técnica algorítmica que lo solucione. Los mini-juegos elegidos para este trabajo son ejemplos de problemas que se resuelven con estos esquemas generales que el usuario ha de aprender, excepto el juego “Adivina la pintura”, ya que éste solo sirve para introducir al jugador en el mundo de los juegos de realidad alternativa en los que se debe buscar información en la web. Describiremos los mini-juegos en apartados posteriores.

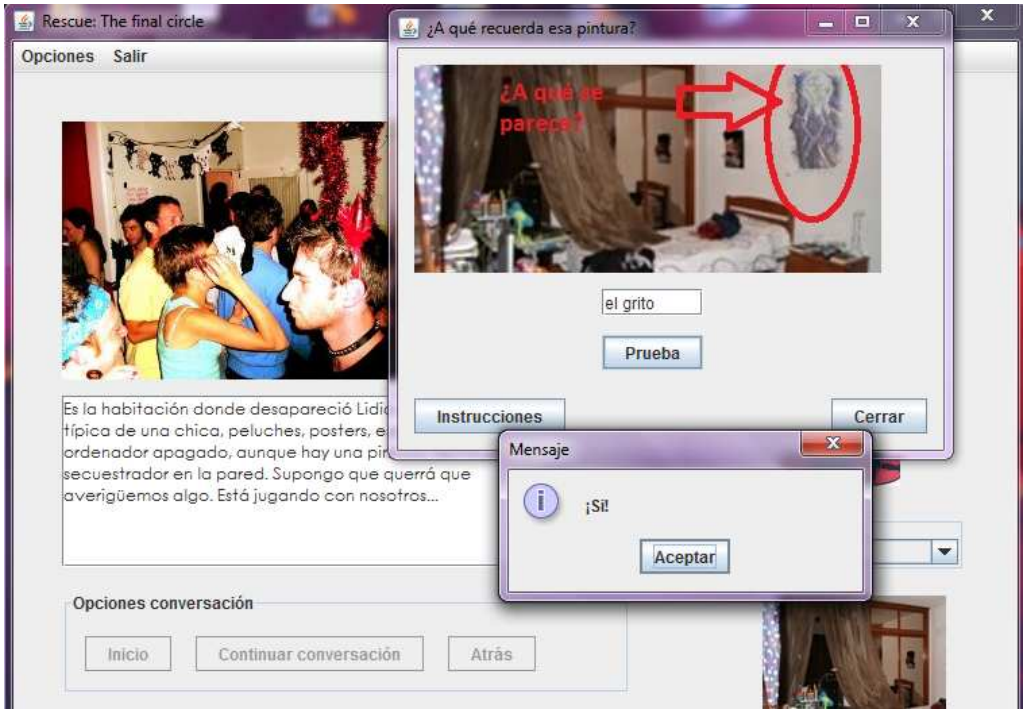

Ilustración 26 Prueba Adivina Pintura

3.3. Planteamiento general

Para realizar el diseño de la aplicación, primeramente hubo que diferenciar tres partes según su funcionalidad.

- **Capa de Presentación: Interfaz gráfica**, sirve de conexión con el usuario final, captando todas las acciones que éste realiza mediante el tratamiento de elementos gráficos básicos (menús, botones, etc.). Asume las funciones relacionadas con la creación de las pruebas lógicas elegidas en el videojuego.
- **Capa de Aplicación: Motor de juego,** es el núcleo del software. Encargado de "construir" las partidas.
- **Capa de Datos: XML,** utilizado para diseñar y almacenar las partidas.

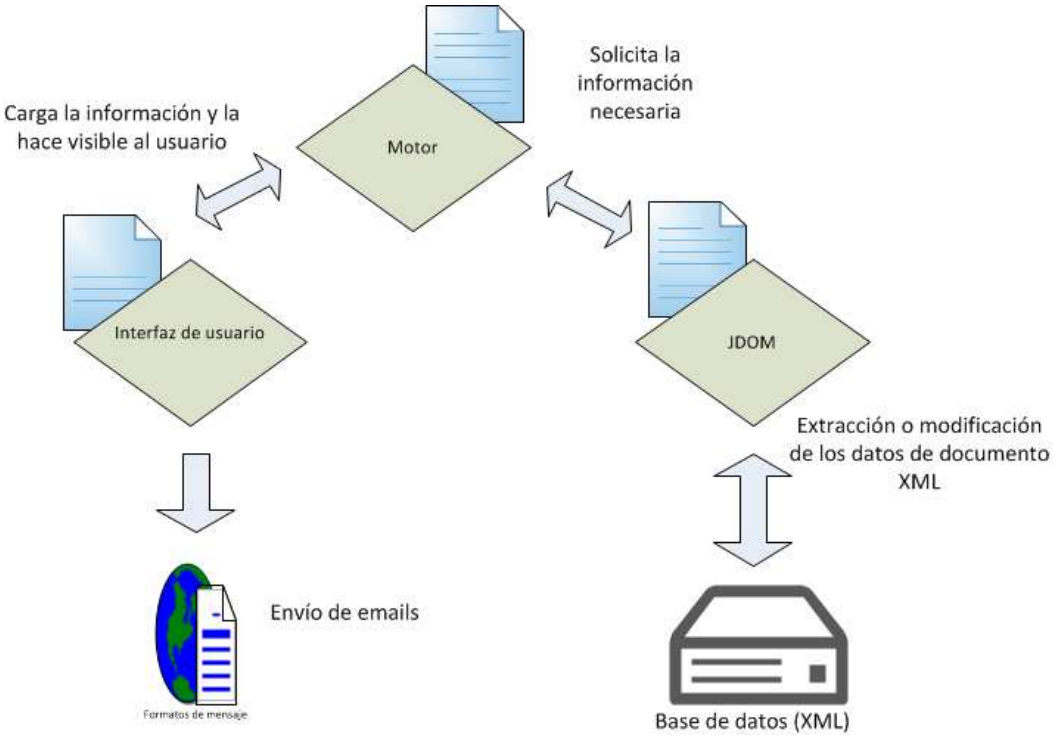

Ilustración 27 Esquema general

Dado que este trabajo consiste en un videojuego educativo, la interfaz gráfica es una parte esencial dentro de este proyecto.

Dicha interfaz tenía que cumplir los requisitos básicos de usabilidad de una aventura gráfico-conversacional:

- ✓ Debía contener los elementos básicos de una aventura gráfico-conversacional, como son los objetos en escena o sus personajes. También era necesario reservar una parte de dicha interfaz para mostrar los diálogos.

- ✓ Debía ser sencilla, para que cualquier usuario pudiera utilizarla sin requerir unas instrucciones. Solo debía ser necesario su propia intuición.
- ✓ La interfaz debía ser capaz de interactuar con el usuario, mostrándole la situación de la partida en un momento concreto y mostrándole los resultados a las distintas peticiones del usuario.

Cada uno de los puntos anteriores nos obligó a adoptar una serie de decisiones de diseño que iremos detallando a continuación. Para conseguir que el software funcione se adoptó una arquitectura basada en eventos. La clase principal es *MostrarGUI*. Esta clase es la encargada de recoger los eventos que se generan cuando el jugador interactúa con la interfaz. Esta clase se apoya en otras clases como por ejemplo la clase *Dialogo* que permite mantener conversaciones con los personajes de la aventura, la clase *Acciones* para poder realizar diversas acciones necesarias en la aventura como coger o usar objetos o la clase *Guarda* que permite guardar y cargar partidas. En el diagrama de clases (figura 49) del anexo B se explica la relación entre las clases.

3.3.1. Capa de presentación: Interfaz gráfica

Cuando se ejecuta el programa por primera vez se pide al usuario que elija entre las opciones de nueva partida, cargar partida anterior o salir del juego.

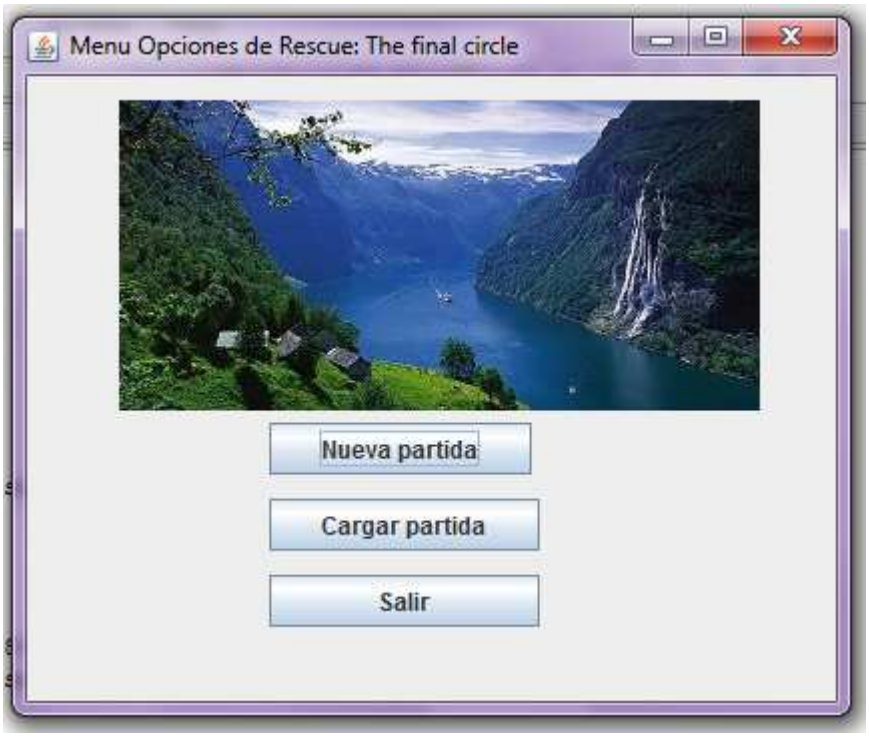

Ilustración 28 Pantalla del Menú de opciones

Si se pulsa el botón “Nueva partida”, saldrá una pantalla en la que Saúl, un chico de unos dieciséis años, explica al usuario que han secuestrado a una

chica. Si el usuario acepta el reto de involucrarse en la misión de rescate, Saúl le pedirá ayuda durante toda la investigación.

Una vez aceptado el reto, el programa pide tu nombre de usuario y una cuenta de correo electrónico. Ya que la información necesaria para recibir las pistas será enviada a dicha cuenta de correo. El programa detecta si el usuario ha introducido una dirección de correo errónea por falta de "@" o ".".

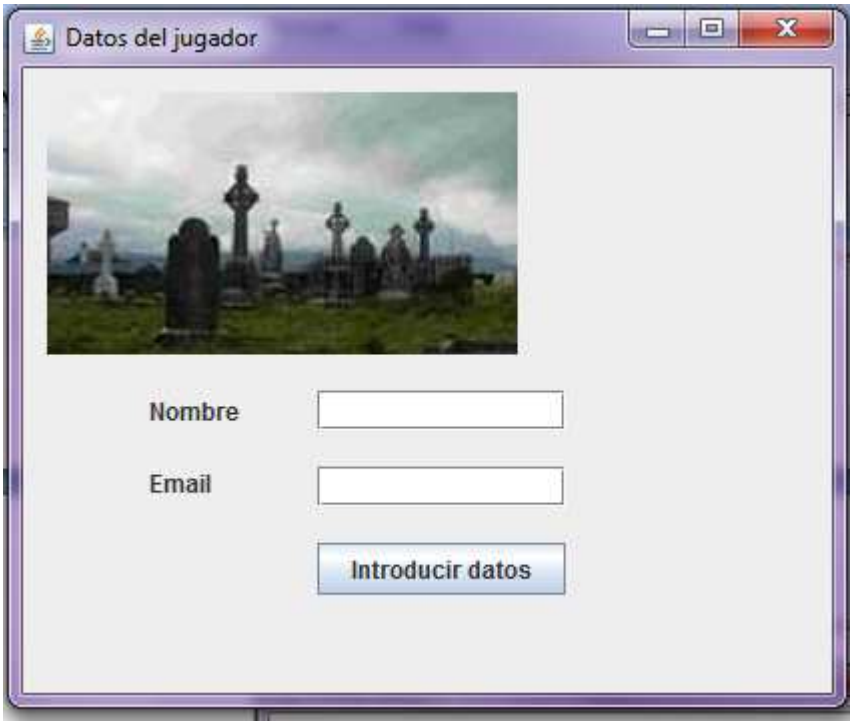

Ilustración 29 Pantalla de petición de datos

Cuando se han introducido los datos pedidos, se mostrará el video que te introduce en la historia: el secuestro de una chica llamada Lidia.


proyectoinicio.mpg

Finalizado el vídeo aparece la interfaz del programa principal en la que Saúl te da detalles más específicos de la historia, además de reproducir un hilo musical que comienza a ejecutarse al detectar el fin del vídeo introductorio y que el jugador puede detener en cualquier momento.

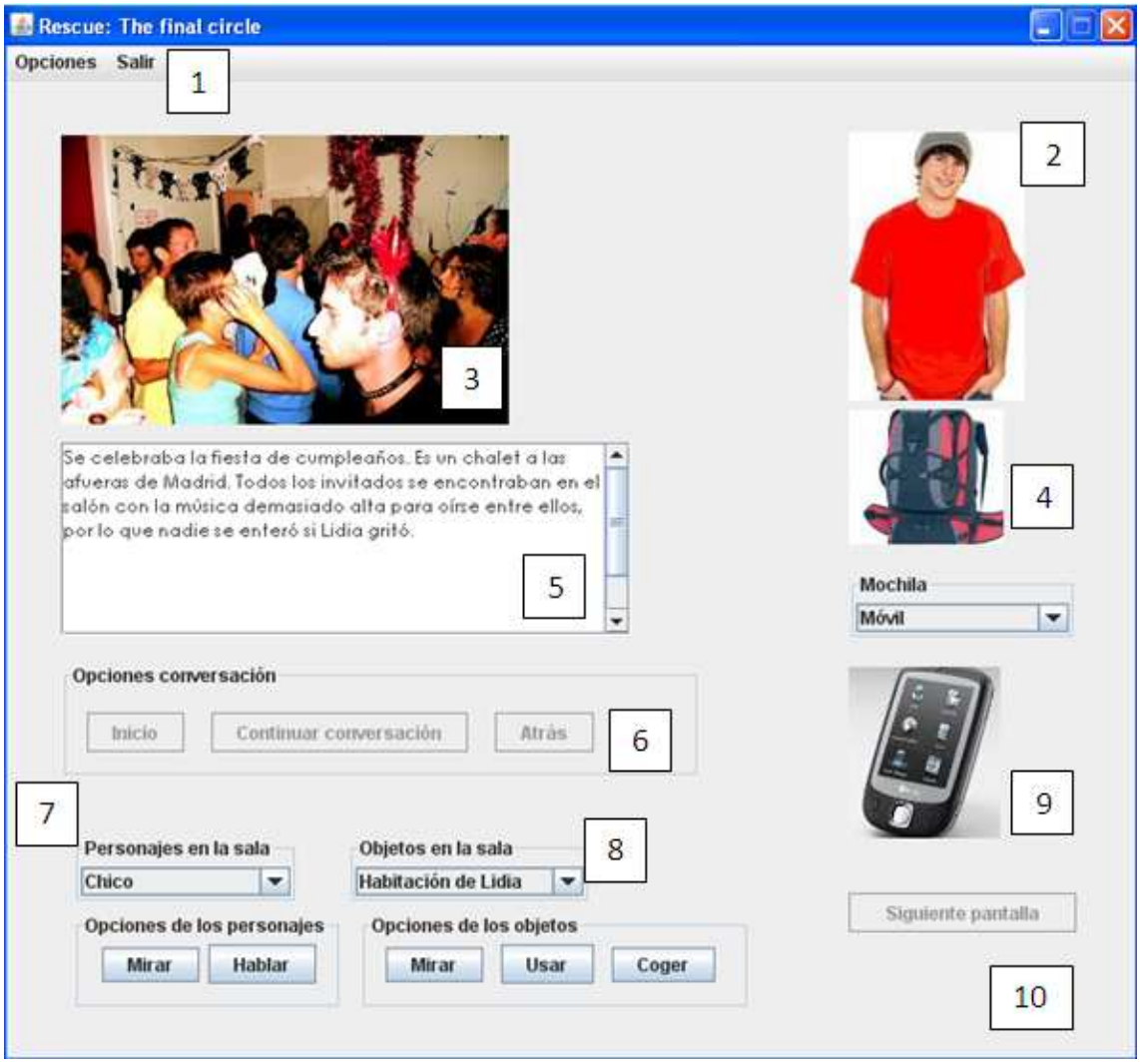

Ilustración 30 Pantalla de una habitación

Los elementos a destacar dentro de la interfaz principal del programa son los siguientes:

1. **Menú**: Mediante el *JMenu* Opciones se pueden realizar las acciones "Guardar partida", "Cargar partida", "Poner sonido" y "Quitar sonido". Seleccionando el menú "Salir" se abandona el juego.
2. **Imagen de los personajes del juego**: *JLabel* que muestra el personaje protagonista del juego. En el caso de hablar con un personaje concreto, mostrará la foto de dicho personaje.
3. **Imagen de la escena**: *JLabel* que muestra la pantalla en la que el usuario se encuentra jugando. Gracias a esta imagen, el usuario puede imaginar el escenario correspondiente donde suceden los acontecimientos.
4. **Inventario de objetos cogidos:** *JComboBox* donde se van almacenando los objetos cogidos durante todas las pantallas. Cuando un objeto se ha usado, desaparece del inventario.

5. **Texto:** *JTextArea* que muestra las descripciones de las escenas, personajes y objetos. También muestra los diálogos posibles con los personajes.
6. **Opciones de conversación con los personajes:** *JButtons* que permiten "Mirar" y "Hablar" con los distintos personajes de una pantalla.
7. **Personajes en la escena:** *JComboBox* que almacena los distintos personajes que hay en una escena. El programa permite mirar y hablar con estos personajes.
8. **Objetos en la escena**: *JComboBox* que almacena los distintos objetos que hay en una escena. El programa permite mirar, coger y usar dichos objetos.
9. **Imagen objetos:** *JLabel* que muestra el objeto elegido en un momento determinado.
10. **Botón siguiente pantalla:** *JButton* que se activa cuando se han cumplido las condiciones para pasar de pantalla. Pulsándolo se pasa a la escena siguiente del juego.

3.3.2. Capa de aplicación: Motor de juego

El motor de videojuego es un elemento esencial dentro de este proyecto. Está compuesto de todo el código necesario para hacer funcionar el videojuego, y tiene en cuenta los vídeos, sonidos, imágenes e inteligencia artificial.

Cuanto mejor esté diseñado un motor, mejor será el juego, ya que las acciones como cargar una escena serán acciones que no consuman demasiados recursos y el usuario no tendrá que esperar demasiado tiempo para cargar una partida o simplemente para hablar con algún personaje o coger un objeto. De esta forma, el usuario podrá centrarse en la historia y no se distraerá por motivos técnicos de software.

El motor de juego realiza siempre las mismas acciones, independientemente de cuántas escenas que tenga el documento XML.

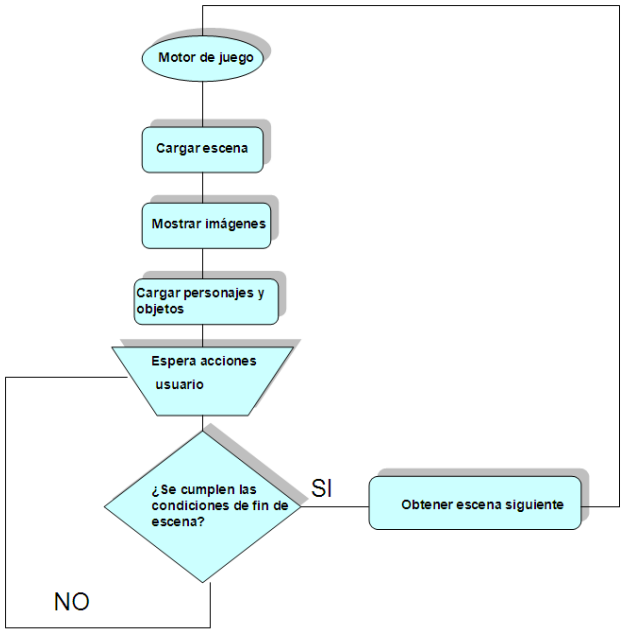

Ilustración 31 Rutinas del motor de juego

El alma de un videojuego es su motor. En nuestro caso, una vez que se arranca el software, el usuario introduce sus datos. Si estamos en el caso de iniciar una partida, o si se carga una partida, el programa ejecuta las rutinas propias. El motor es el encargado de crear y mostrar la interfaz con la que el usuario tendrá que interactuar. La clase *Motor* creará una instancia de la clase *MostrarGUI*, la interfaz de usuario, la cual será la encargada de recoger las acciones que el usuario realice, tales pulsar el botón "Coger" para almacenar utensilios en el inventario o hablar con un personaje. En definitiva, recogerá las acciones que el usuario realizará sobre el juego y acciones que deberán realizarse en el instante adecuado. Las acciones que llevan a cabo son:

a) Detección del fin de una escena

La clase *MostrarGUI* detectará el fin de una escena, es decir, cuando los requisitos a cumplir dentro de una pantalla se han completado, se activa el botón "Siguiente Pantalla" que permite cargar la siguiente pantalla.

b) Detalles de la pantalla

Durante una pantalla concreta, el jugador necesita obtener la descripción de la escena en la que se encuentra, acompañado de una fotografía de la ubicación del personaje, mostrando así más detalles. Se pensó en diseñar esta parte de la pantalla así para favorecer la inmersión del jugador en la historia. Las reglas a seguir dentro de la interfaz se explican en el manual de usuario en el anexo C de este

documento. Es necesaria la intuición del jugador, ya que son las reglas básicas de cualquier aventura conversacional: por ejemplo, antes de hablar con un personaje, el usuario ha tenido que mirarlo. La razón es que al pulsar el botón "Mirar" aparece la descripción correspondiente. Dicha descripción puede aportar información valiosa para la aventura. Sucede lo mismo en el caso de coger objetos, hay que haberlos mirado antes. Si el usuario quiere coger un objeto, el programa muestra un mensaje cuando se cogen o cuando no se puede realizar la acción. Cuando dos objetos se pueden usar puede aparecer un nuevo elemento en la habitación o puede dar lugar a la aparición de una prueba lógica que el usuario debe completar satisfactoriamente para avanzar en la investigación.

c) Recursos de una escena

Para mostrar las imágenes tanto de la sala como de los personajes y objetos, se ha creado la clase *Lector*. Esta clase, como su nombre indica, es capaz de leer un documento XML y obtener los elementos adecuados a las necesidades. Es decir, en una escena en concreto, como por ejemplo la escena "La fiesta", la clase *MostrarGUI* deberá cargar las imágenes de la escena y de los personajes, junto con los sus diálogos, objetos y aquellos objetos que forman parte del inventario. La clase *Lector* es la encargada de obtener toda la información para completar la escena que va a cargar la clase *MostrarGUI*.

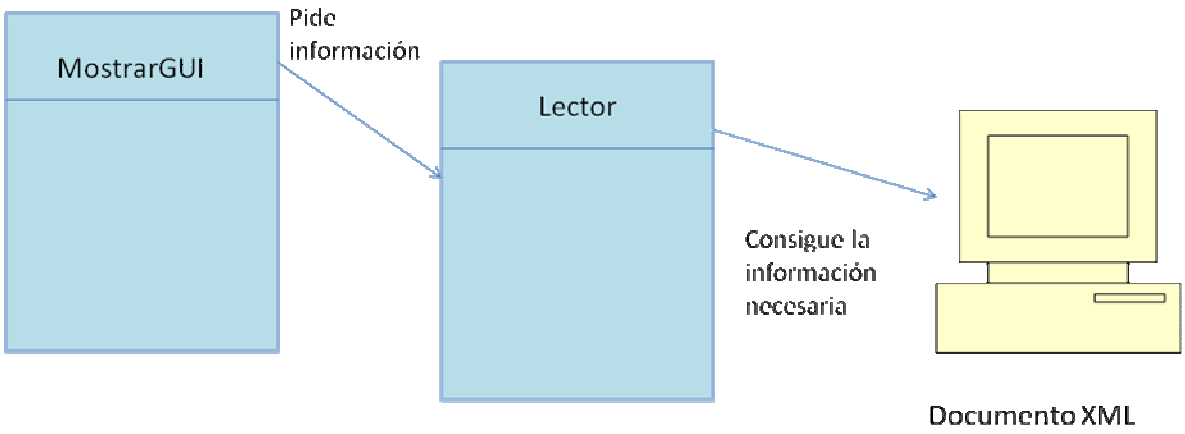

Ilustración 32 Esquema de obtención de datos

d) Diálogos

La encargada de hacerlo es la clase *MostrarGUI* junto con la clase *Dialogo*, que obtiene los diálogos a través de métodos de la clase *Lector*. Para poder hablar con un personaje hay que pulsar el botón

"Hablar" de la interfaz gráfica. Una vez pulsado aparecerá en la parte de la interfaz reservada para texto, varias opciones de diálogo (entradas de diálogo), es decir, diversas frases que el usuario puede elegir para hablar con el personaje. Es necesario hablar con los personajes para obtener información relevante en la historia, pero no todas las entradas llevan a la obtención de la información correcta. Para que el usuario pueda elegir qué decir a un determinado personaje, la forma de mostrar la petición de elección de una determinada entrada se realiza usando *JDialog*.

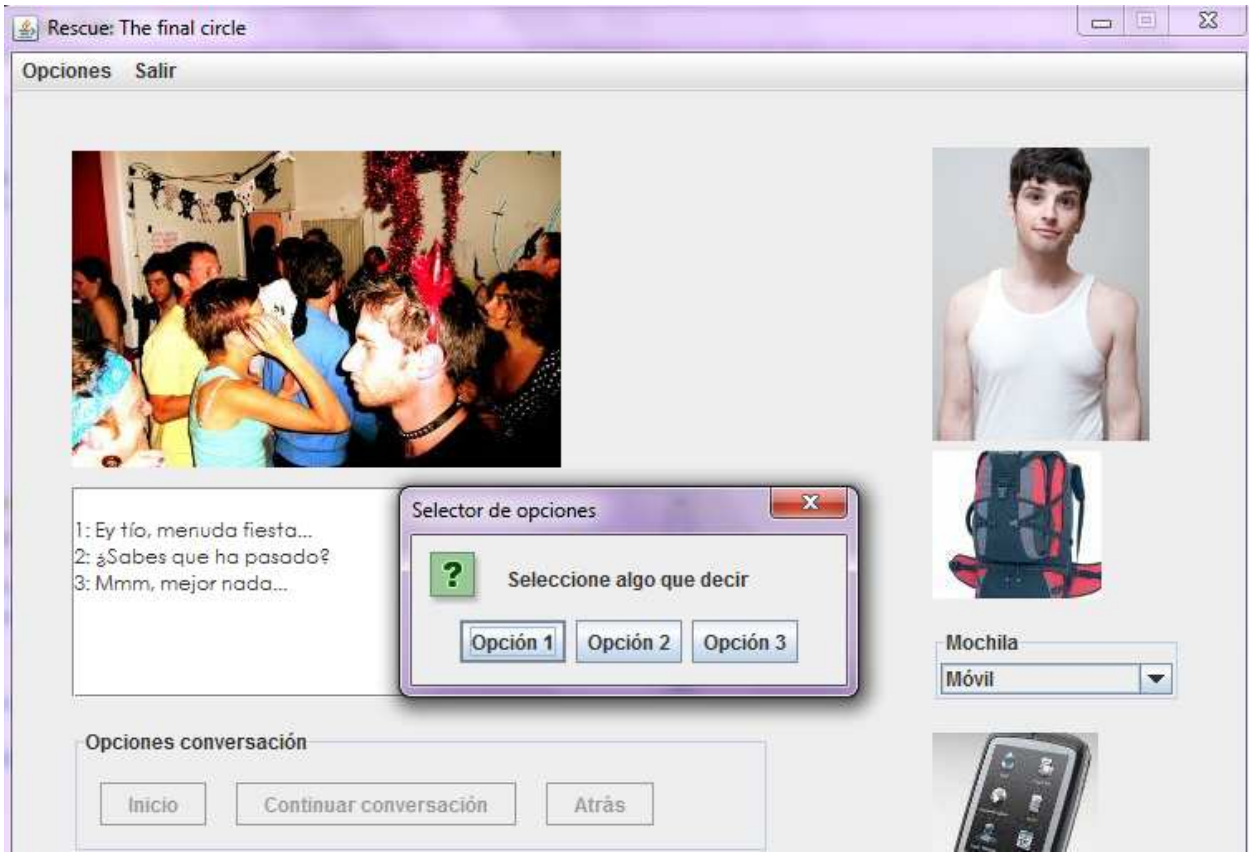

Ilustración 33 Selector de opciones de diálogo

Se decidió seguir esta forma de gestionar los diálogos para no introducir más elementos en la pantalla principal del videojuego. Desde un principio se tuvo claro que la interfaz tenía que ser sencilla para el jugador. Además es una forma sencilla de pedir al usuario la frase a decir. Durante todo el programa, es el jugador quien elige cuando hablar con un personaje, nunca comienza un personaje del juego, por tanto, el *JDialog* (llamado "Selector de opciones") solo aparece en pantalla cuando el usuario quiere hablar con un personaje concreto. En el selector de opciones aparecen tantos botones de opción como entradas de diálogo haya en el diálogo concreto. Una

vez pulsado el botón de la opción correspondiente, se activan las siguientes opciones de conversación:

1. **Inicio**: El botón inicio sirve para iniciar la conversación que se está manteniendo en ese momento con el personaje. Es posible que el jugador no llegue a la respuesta deseada, y con el fin de que no tenga que volver a repetir los pasos seguidos para hablar con el personaje, se incluyó este botón, que hace más sencillo el inicio de la conversación.
2. **Continuar conversación:** Este botón tiene la misión de progresar en la conversación. Una vez pulsado el botón de la opción elegida se muestran los diálogos intercambiados entre el jugador y el personaje de la escena. Para avanzar se pulsa en el botón "Continuar conversación" volviendo a cargarse las nuevas entradas de diálogo y el Selector de opciones.
3. **Atrás:** Este botón sirve para mostrar diálogos anteriores al diálogo actual. Si el jugador se da cuenta de que la entrada de diálogo elegida no le conduce a la obtención de información útil, no es necesario volver a iniciar el diálogo. Pulsando el botón "Atrás" se vuelven a cargar las entradas de diálogo que se habían mostrado en un paso anterior, dando la oportunidad al jugador de elegir otra conversación. Este botón no es propio de las aventuras conversacionales, en las que hay que volver a hablar con el personaje de la escena y que en muchas ocasiones, es algo muy aburrido para el usuario, ya que tiene que hacer el esfuerzo de releer algo que ya conoce.

e) Objetos

Una característica significativa y típica de las aventuras conversacionales y gráficas es la utilización de objetos que el jugador va encontrando por diferentes escenarios. En este juego para poder usar los objetos que están en la sala el jugador tiene que mirarlos para obtener su descripción y coger aquéllos que considere necesarios para una acción futura. Para poder usar dos objetos es necesario haberlos cogido y almacenarlos en el inventario o usar un objeto cogido con un objeto de la sala donde se encuentra el jugador. Para realizar este tipo de acción, la clase *MostrarGUI* se apoya en la clase *Acciones* y *GestiónObjetosInventario*.

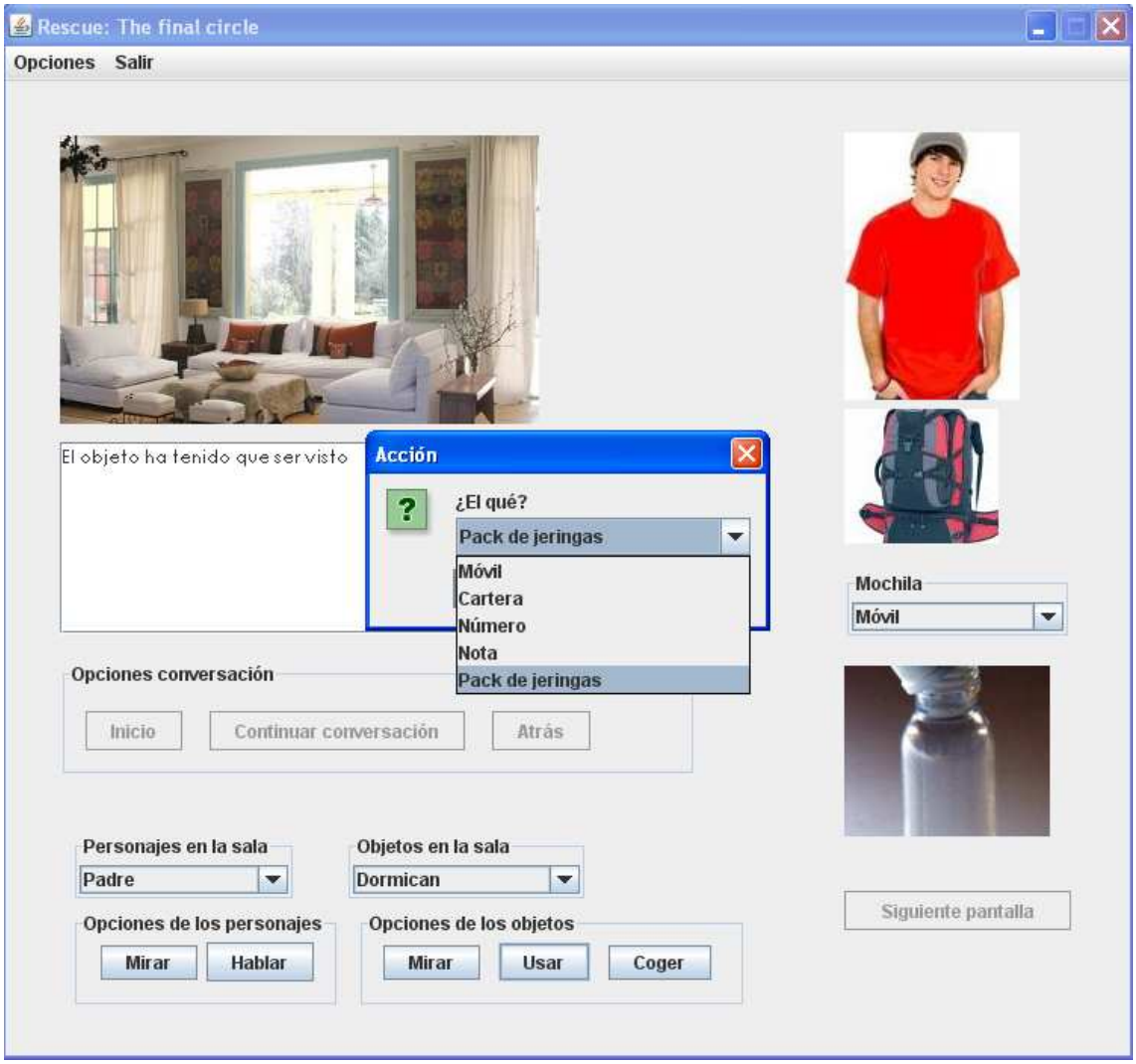

Ilustración 34 Acción usar objetos

f) Pistas

La clase *MostrarGUI* también es la encargada de gestionar el envío de pistas al usuario a través de la clase *Mail* ya que *MostrarGUI* detecta el momento correcto para el envío y lo envía a través de la clase *Mail* en la que se apoya. Las pistas sirven para evitar que el usuario quede atascado en una escena y para simular que un personaje existe en la vida real. En todos los ARGs se utiliza este método para involucrar al jugador en la historia que se narra. Por ejemplo, en el ARG "Inmemorian II", el jugador forma parte del equipo encargado de dar con un asesino en serie. En este juego se reciben emails por parte del equipo de investigación, e incluso, del propio asesino.

En nuestro caso, el jugador solo recibe emails de Saúl, ya que Saúl es el encargado de moverse por aquellos escenarios donde se van encontrando pistas de Lidia. El usuario recibirá la información que Saúl va obteniendo.

Para evitar que el usuario se desmotive y no consiga pasar de pantalla, se envían pistas cuando:

- ✓ El usuario ha fallado en cuatro ocasiones la solución del mini-juego: el programa detecta el número de veces falladas y como consecuencia envía una pista al usuario.
- ✓ El usuario no ha resuelto el min-juego en 30 minutos: el programa detecta que el usuario no ha pasado de escena.

g) Consecuencias a los eventos

La interfaz de usuario es la encargada de recoger los eventos y de dar lugar a las consecuencias de éstos, que en muchos casos, son la aparición de una prueba. Cada de una de estas pruebas de lógica aportarán habilidades al usuario en la resolución de problemas de un tipo concreto de técnica.

La lista de pruebas a resolver por el jugador es la siguiente:

- **Círculo**

En esta prueba se pide al usuario que elija la posición en la que sus amigos Saúl y Lidia deben colocarse dentro de un círculo. Es un círculo formado por 41 personas, en la que irán muriendo una de cada tres, empezando por la persona que está en la primera posición. Este juego pertenece a la técnica de resolución por fuerza bruta, aunque también puede clasificarse como resta y vencerás. Consiste en la resolución de un

problema probando todas las posibles combinaciones. Es decir, para poder resolver este juego es necesario que el jugador vea todas las posibles posiciones donde deben colocarse Saúl y Lidia para no morir. Resolviendo este problema el usuario aprenderá la forma lógica en la que funciona, por ejemplo, un ordenador.

- **Clave Museo**

El jugador se encuentra una nota cifrada al llegar a la sala de un museo. Para poder continuar la búsqueda de Lidia es necesario descifrar dicha nota. Este juego pertenece a la técnica de transforma y vencerás. Solo consiguiendo la clave que permite conocer cuál es la transformación a realizar, se podrá continuar con la investigación.

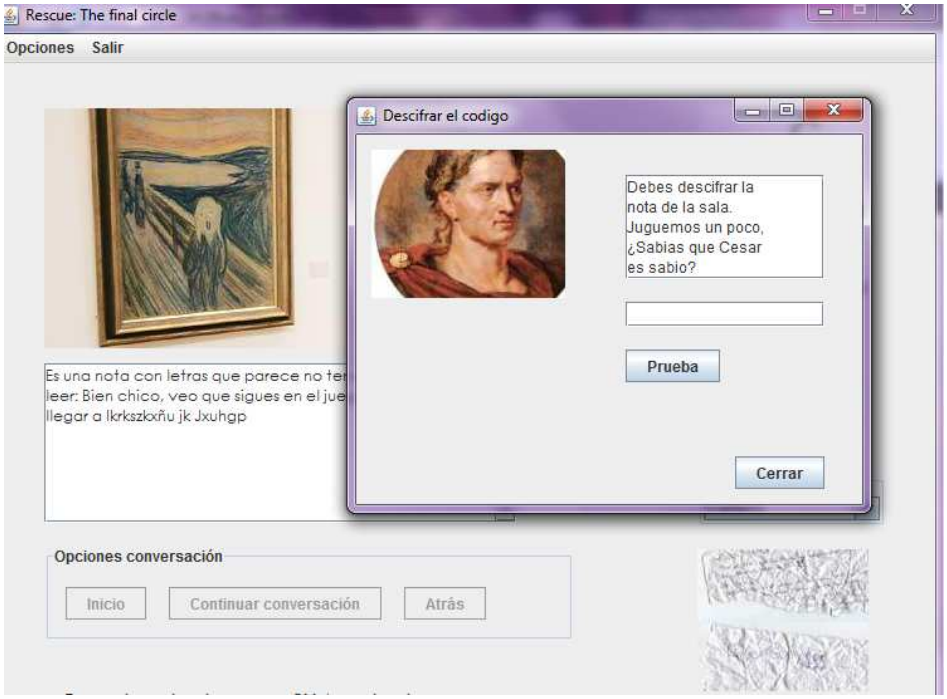

Ilustración 35 Descifra la clave

- **Cruzar fiordo**

Esta prueba está sacada del famoso juego en el que un pastor debe cruzar al otro lado del río con un lobo, una oveja y con la comida de la oveja, sabiendo que existen determinadas combinaciones que están prohibidas. Cruzar el fiordo pertenece a la técnica de resta y vencerás.

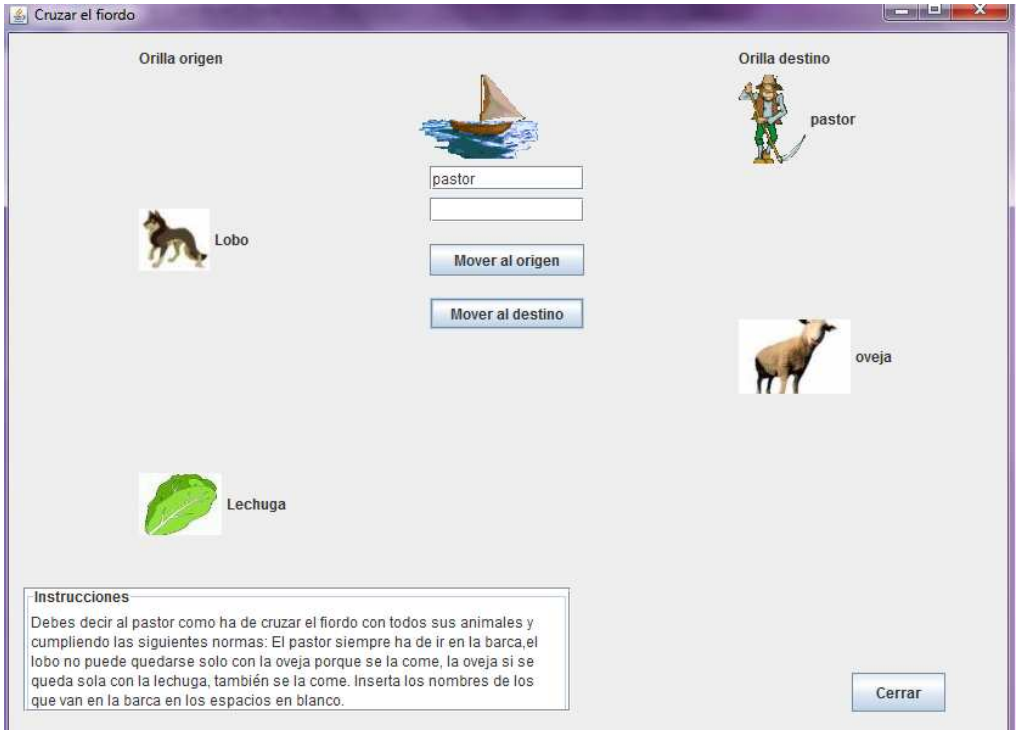

Ilustración 36 Cruza fiordo

- **Dormir al perro**

Este juego consiste en obtener 4ml a partir de tres recipientes que miden 8ml, 5ml y 3ml. El problema surge al tener la necesidad de tener 4ml pero dichos recipientes no tienen ningún tipo de medida y solo se conoce la capacidad de cada uno. En este problema se aprende el concepto de "variable auxiliar" muy usado en programación, ya que para mover el líquido de un recipiente a otro es necesario almacenar una parte en un recipiente adicional.

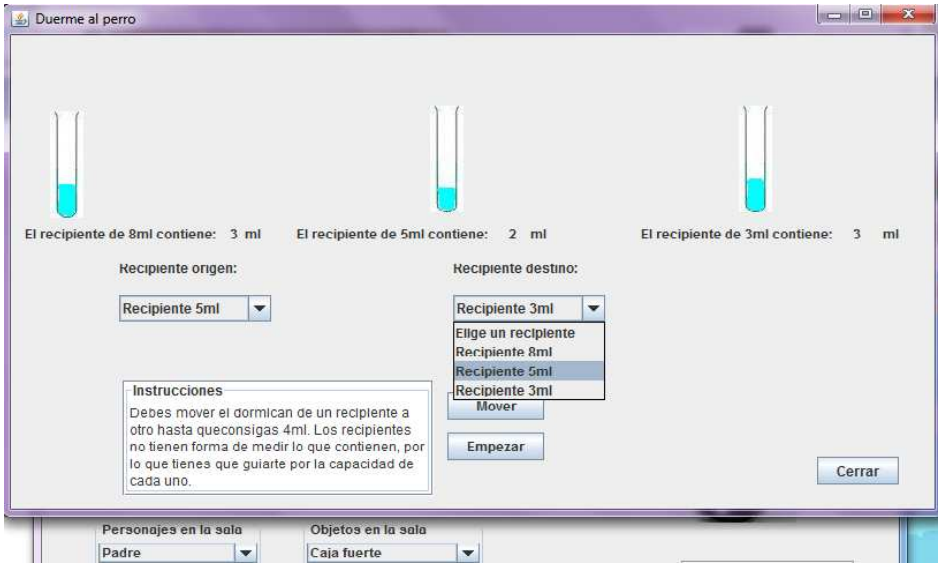

Ilustración 37 Duerme al perro

- **Encuentra la tumba**

En este problema se busca que el usuario empiece a conocer secuencias famosas. Todo programador novel ha comenzado programando series como la de los números primos o sucesiones como la de *Fibonacci*. En un futuro será más fácil aprender los algoritmos famosos si se va aprendiendo secuencias de este estilo.

Ilustración 38 Encuentra la tumba correcta

- **Números enigmáticos**

En este juego un personaje en un cementerio nos pide acertar un número de tres cifras en diez intentos. Obviamente el jugador debe seguir una serie de pasos para poder llegar a la solución. Esa serie de pasos que el jugador debe aprender es el algoritmo de la búsqueda binaria. Lo primero de todo es tomar el elemento del centro. El personaje del cementerio nos dirá si el número que decimos es mayor o menor. Si el elemento buscado es menor, tomamos el intervalo que va desde el principio al elemento central, en caso contrario, tomamos el elemento que va desde el elemento central hasta el final del intervalo.

Procediendo así los intervalos se nos van reduciendo hasta que encontremos el número pedido.

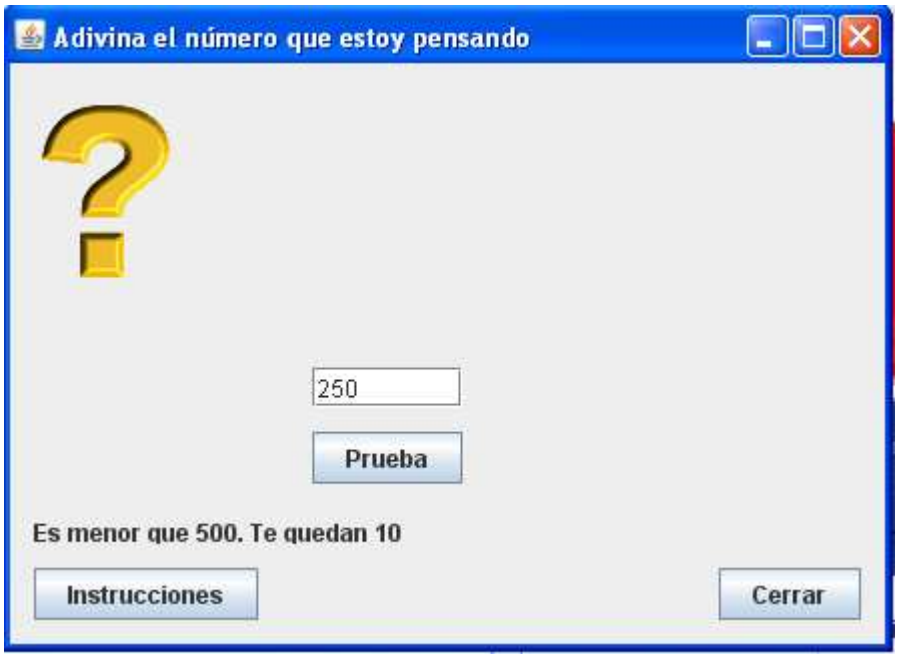

Ilustración 39 Adivina el número

El número a adivinar es el 124. Se llega a esta cifra después de aplicar diez veces el algoritmo correspondiente a este juego.

- **Sacar dinero del cajero**

Esta prueba consiste obtener el pin de un cajero para sacar dinero. Hay que adivinar el número que sigue a una secuencia.

El jugador debe averiguar la relación que siguen los números dados para poder conocer el siguiente número. En este caso los números siguen un orden alfabético si se tratan como un texto en lugar de cómo número. Esta prueba pertenece a la técnica de transforma y vencerás.

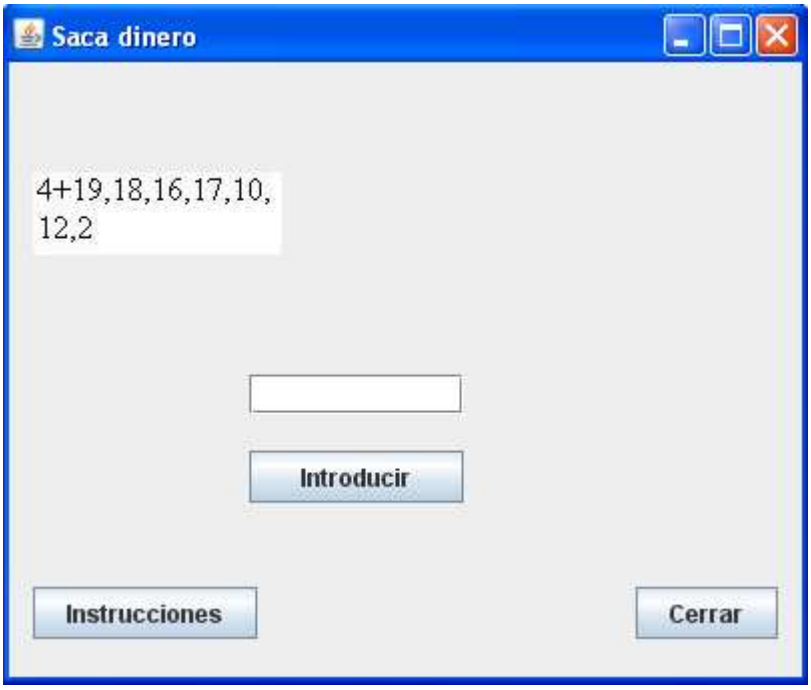

Ilustración 40 Sacar dinero del cajero

- **Vitrinas**

En este problema hay que pasar unos platos de arte maya de una estantería a una estantería de una exposición. Cómo son platos pertenecientes a un museo, no deben romperse. Los platos están apilados de tal forma que el plato de mayor radio está situado en la parte inferior de la pila mientras que el menor en la parte superior. Para realizar esta tarea tenemos una vitrina en la que podremos ir apoyando los platos según la necesidad del momento. Para resolver este problema hay que utilizar lo que se conoce como recursividad.

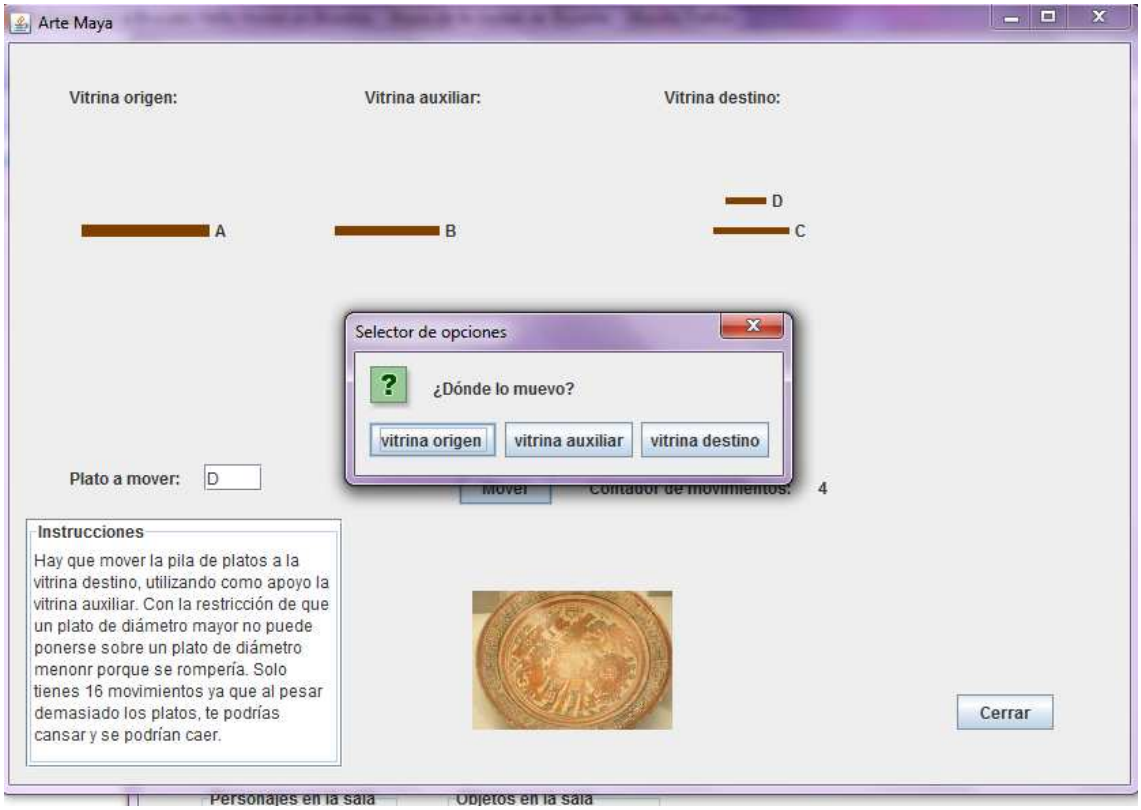

Ilustración 41 Vitrinas

En este juego si se intenta hace un movimiento prohibido, se avisa al usuario que no se puede realizar. Además, para superar esta prueba, se ha puesto un contador de movimientos, con el fin de que el usuario lo haga en el menor número de pasos posibles para una “torre” de cuatro platos, ya que si lo realiza así, nos aseguramos de que está realizando los movimientos en el orden correcto. Este problema se resuelve en 15 movimientos y al jugador se le permiten 16.

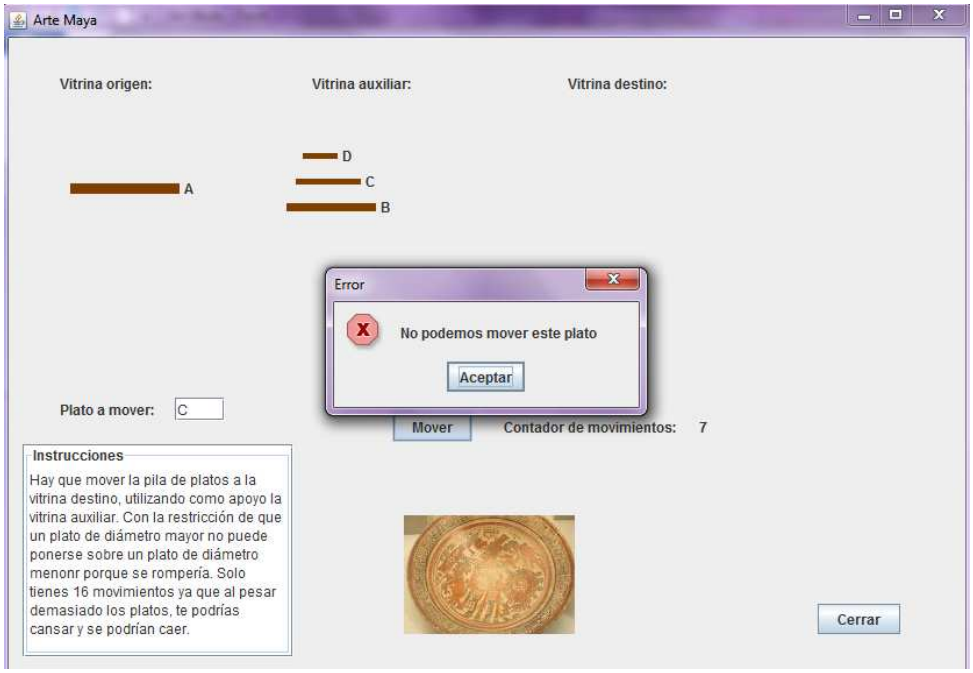

Ilustración 42 Movimiento prohibido

3.3.3. Capa de datos: JDOM y documentos XML

Esta capa se encarga de almacenar los datos de forma ordenada y formal, de tal forma que su recuperación y actualización se pueda hacer de forma sencilla.

La capa de datos, como se ha comentado anteriormente, guarda los datos del estado de la partida para el correcto funcionamiento del videojuego. Estos datos son manejados por la capa de aplicación y mostrados al usuario mediante la capa de presentación.

Cuando se pensó en cómo realizar esta capa, hubo que decidir entre dos opciones válidas. La primera consistía en la utilización de una base datos y la segunda almacenando los datos en ficheros. La decisión final fue guardar los datos de las partidas en documentos XML. [39]

Se descartó la primera opción ya que dicha forma se realiza normalmente cuando hay grandes cantidades de datos a almacenar y los sistemas de búsqueda en estas grandes bases de datos son más eficientes que las búsquedas en ficheros. La ventaja que tiene la utilización del lenguaje XML, entre otras, es que los datos los almacena de forma jerárquica, lo que facilita la búsqueda de cualquier dato sin tener que ir línea a línea.

XML es un lenguaje de marcas que ofrece un formato para la descripción de datos estructurados, el cual conserva todas las propiedades importantes de SGML (“Lenguaje de Marcas Estándar Generalizado”). XML es un metalenguaje, dado que con él se puede definir el lenguaje de presentación y se centra en la información contenida. No posee etiquetas y el diseñador puede diseñarlas según su propio criterio. De esta forma se da mucha libertad al diseñador.

Para poder dar la libertad suficiente al desarrollador es necesario un "diccionario" de los elementos que van a existir en el documento XML. La estructura de los ficheros XML vienen dada por la definición encontrada en las DTDs correspondientes al fichero XML.

Características del lenguaje XML:

- ✓ Es fácil la escritura de programas que procesan la información de los documentos XML.
- ✓ Los documentos XML son fácilmente legibles e intuitivos.
- ✓ XML es simple pero perfectamente formalizado.
- ✓ Puede convivir con otros lenguajes, como Java, JavaScript, HTML, Visual Basic, etc.
- ✓ Estructura en forma de árbol, lo que permite crear una jerarquía entre los distintos elementos que forman una escena. Por ejemplo, dentro de una escena, existen elementos "PERSONAJE", que están al mismo nivel que los elementos "OBJETO". Los elementos "PERSONAJE" tienen elementos hijo "DIALOGO", que a su vez, tienen el elemento hijo "ENTRADA DE DIALOGO".

En el siguiente gráfico se verá con más claridad la jerarquía que se decidió que tuviera el documento XML:

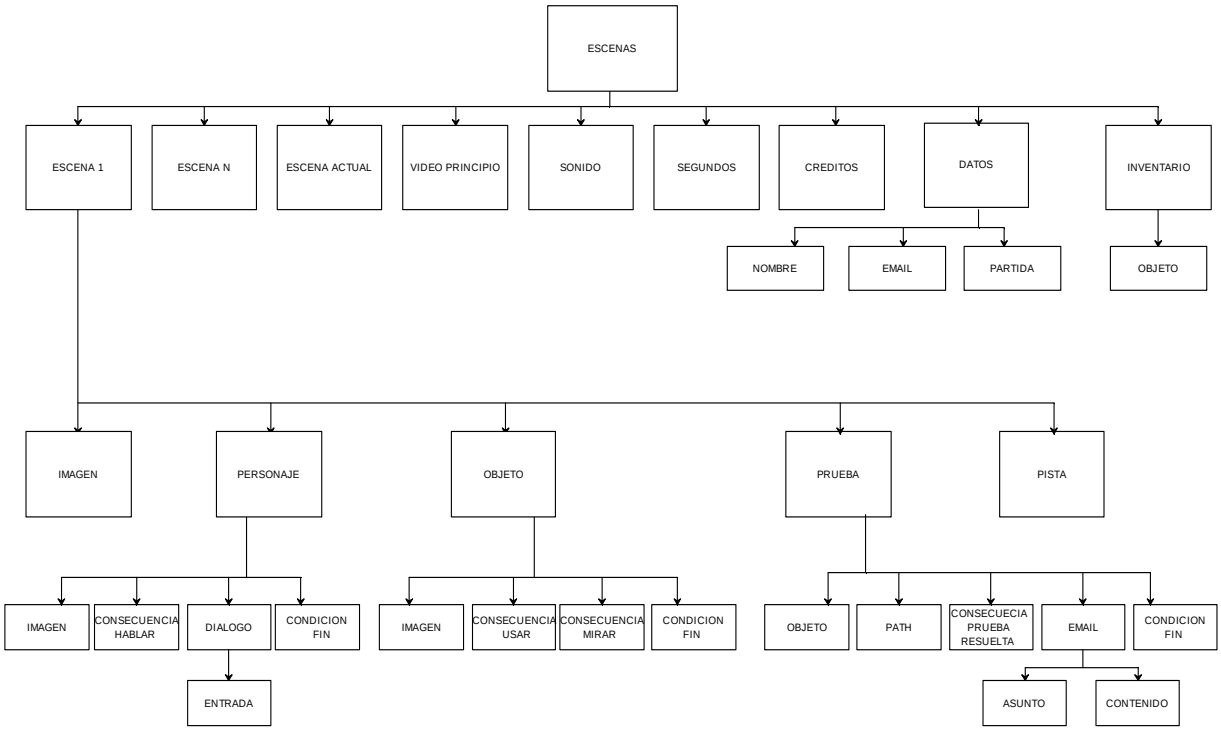

Ilustración 43 Jerarquía documento XML

- ✓ Soporte cómodo para guardar el estado de las partidas en memoria. Los ficheros XML, son almacenados en el propio disco duro de la máquina en la que se está ejecutando el videojuego, puesto que al no tratarse de una aplicación Web cliente-servidor y al no ser una aplicación que tenga que tener un alto grado de privacidad, la seguridad no es un requisito indispensable.
- ✓ Fácilmente legible por las clases encargadas de ello, en nuestro caso, por la clase *Lector*.
- ✓ Es extensible. Después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna.
- ✓ Transformamos datos en información, pues se le añade un significado concreto y los asociamos a un contexto, con lo cual tenemos flexibilidad y sencillez para estructurar documentos. Admite cualquier nomenclatura de los diversos campos, por lo que podemos diseñar con mucha libertad las escenas del videojuego, es decir, de cuantas escenas constará el videojuego, que elementos contendrá y las características como su descripción y por ejemplo, condiciones tales como si son los objetos son visibles de primeras o no.

Desde que se pone en funcionamiento el software se trabaja con documentos XML. Cuando se arranca el videojuego se presenta al usuario una pantalla con diferentes opciones: “Partida Nueva”, “Cargar Partida” o “Salir”.

Las dos primeras opciones requieren de un fichero XML, la diferencia entre ellas es la carga de un fichero distinto en cada caso: la primera opción carga una partida sin los elementos modificados. Cuando el usuario interactúa con la interfaz, se van guardando esas modificaciones en el árbol JDOM, el cual, guardará los cambios realizados en un nuevo documento XML si se elige la opción de guardar la partida. La segunda opción, cargar partida, simplemente carga una partida que ha sido guardada con anterioridad, reflejando los cambios realizados hasta dicho momento.

3.3.3.1. Procesar documentos XML

Los documentos XML se analizan por medio de procesadores, módulos de software que permiten leer los documentos XML y proporcionan acceso a su contenido y estructura.

Existen procesadores válidos que comprueban las restricciones de validez y de buena formación que se definen en la recomendación de XML y muestran las violaciones y errores que puedan existir. Los procesadores no válidos solo comprueban si el documento está bien formado.

Para procesar un documento XML después de acceder a su estructura interna, es conveniente utilizar una API (Interfaz de Programación de

Aplicaciones). A continuación mencionaremos tres APIs que se utilizan en procesadores XML:

3.3.3.2. Modelo de Objeto de Documento (DOM)

Es un API basada en una estructura en forma de árbol jerárquico. En DOM, un documento viene representado como un árbol con ramificaciones en forma de elementos. DOM proporciona un conjunto de servicios para poder acceder y manipular las ramificaciones del árbol. Un procesador XML basado en DOM crea en la memoria la estructura de un documento XML, de forma que puedan utilizarla las aplicaciones que trabajen con el documento. El uso de DOM está indicado en los siguientes casos:

- ✓ Cuando se modifica estructuralmente un documento XML, al ordenar los elementos de una determinada forma o al desplazar algunos elementos dentro del documento.
- ✓ Cuando se comparte el documento en memoria con otras aplicaciones. DOM define un conjunto de interfaces Java para crear acceder y manipular estructuras internas de documentos XML, también define una interfaz de plataforma de lenguaje neutral para programas de aplicación en forma de un conjunto estándar de objetos. La forma en que funciona DOM consta de dos fases: primero se analiza un documento XML y después se accede al árbol DOM.

Las ventajas de este API son:

- ✓ Se puede añadir un nodo al árbol en cualquier punto.
- ✓ Puede eliminarse cualquier nodo del árbol.

Por el contrario, las desventajas son las siguientes:

- ✓ Se crean árboles demasiado grandes que pueden ocupar mucho espacio.
- ✓ Para obtener información específica del árbol se debe recorrer desde el inicio.
- ✓ No fue creado para ningún lenguaje de programación en concreto, lo que dificulta su uso al utilizar distintos lenguajes como pueden ser Java o *C++* entre otros.

3.3.3.3. API Simple para XML (SAX)

Es una API que está orientada a eventos. Un procesador XML con SAX no genera estructura de datos, lo que genera son eventos, como por ejemplo el evento principio de un elemento y el final del otro, de acuerdo a una entrada del documento XML con el que se está trabajando. El programa de aplicación capta estos eventos y ejecuta lo que corresponde a la tarea que está efectuando, como puede ser obtener los valores de los atributos. SAX

resulta de cierta forma más eficaz que DOM ya que no crea una estructura de datos explícita.

Resulta conveniente:

- ✓ Cuando se trabaja con documentos largos que no caben en memoria.
- ✓ Cuando se ejecutan tareas en los elementos que no afectan la estructura global del documento o se extrae el contenido de un elemento específico.

Las ventajas son las siguientes:

- ✓ Ideal para manipular archivos de gran tamaño
- ✓ Es más rápido y sencillo de utilizar que DOM
- ✓ Manejo de eventos

Por el contrario, los inconvenientes son:

- ✓ Una vez procesada la información no es posible volver a obtenerla, a menos que se repita el proceso.
- ✓ Fue creado para cualquier lenguaje de programación.

3.3.3.4. JDOM *(Java Document Object Model)*

JDOM es un API para leer, crear y manipular documentos XML. Trabaja de forma similar a DOM por lo cual hereda las funciones básicas de esta API, siendo la principal diferencia.

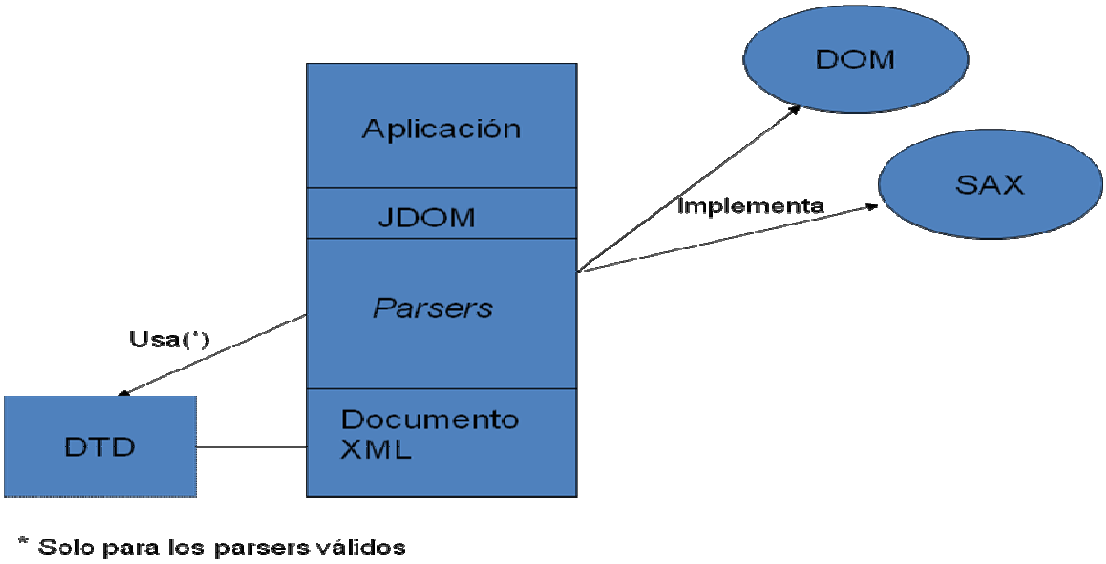

Ilustración 44 Esquema general de XML

Las ventajas de usar este API son:

- ✓ JDOM fue hecho pensando en Java, por esta razón la facilidad de utilizarlo.
- ✓ Es el más sencillo de usar de todas las APIs mencionadas anteriormente.

La desventaja es:

- ✓ Crea un árbol con los datos tal y como lo hace DOM.

A continuación muestran las ventajas y desventajas de las APIs:

	Sencillez	**Requerimiento de recursos**	**Recuperación de la información**	**Conjunción con Java**	**Manejo de eventos**
DOM	Malo	Malo	Bueno	Malo	Malo
SAX	Malo	Bueno	Malo	Malo	Bueno
JDOM	Bueno	Malo	Bueno	Bueno	Malo

Ilustración 45 Resumen de características de los APIs

Las partidas son documentos XML con estructura de árbol jerárquico. Por ésta y por las razones mencionadas anteriormente se ha elegido utilizar JDOM en este trabajo frente a las otras dos APIs. JDOM era de gran utilidad al programar el código en Java: con ella recuperaremos la información, mediante la clase *Lector*, para poder crear y/o actualizar la interfaz de usuario con los datos obtenidos del documento.

3.4. Pruebas de funcionalidad y usabilidad

Toda aplicación debe ser probada para ver su robustez y conocer si gusta al cliente. La fase de pruebas debe estar planificada desde un principio y se debe hacer un seguimiento hasta que el cliente da su aprobación. Se debe empezar desde “lo pequeño” y progresar hasta “lo grande”. [37]

En proyectos de la vida real, la fase de pruebas la realiza un equipo independiente, el cual, buscará de forma concienzuda los fallos de software.

En este trabajo las primeras pruebas realizadas fueron las pruebas conocidas como pruebas de unidades. Se realizaron sobre pequeños módulos del proyecto, para asegurarse de su funcionamiento independientemente del resto del trabajo. Así era más sencillo detectar ciertos errores antes de probar el funcionamiento del sistema. Durante el desarrollo del videojuego se realizaron muchas pruebas.

Vamos a describir aspectos a tener en cuenta a la hora de validar:

1. Pruebas relacionadas con los objetos:

   a) Si el jugador elige la opción “Coger” un objeto, este objeto debía haber sido visto anteriormente, con el fin de que el jugador pudiera haber leído la descripción de este objeto, que es posible que resultara interesante en escenas posteriores.
   b) Detectar que se ha cogido un objeto, eliminándolo de la lista de objetos en la sala y pasándolo al inventario.
   c) Si al usar dos objetos se carga un mini-juego y éste se cierra sin haberse resuelto, los objetos deben mantenerse en la posición inicial, ya que deben volver a usarse posteriormente.

2. Pruebas relacionadas con los diálogos:

   a) Debe comprobarse que se muestran tantas opciones de diálogo a la hora de elegir los diálogos como diálogos posibles con ese personaje.
   b) Debe poder irse hacia un diálogo anterior por si el usuario quiere elegir otra entrada de diálogo, ya que ese camino podría no conducir a la información requerida.
   c) Comprobar que el diálogo corresponde al personaje con el que está hablando si hay varios personajes en la escena.

3. Pruebas relacionadas con los vídeos del videojuego:

   a) Detectar el fin del vídeo para cargar la pantalla principal a pesar de que la duración cambie.
   b) Detectar el fin del vídeo al acabar el videojuego para que se cierre el proceso correctamente.

4. Pruebas relacionadas con el envío de pistas:

   A la hora de enviar pistas al usuario hubo que tener en cuenta dos casos:

   a) Envío de pista si se falla cuatro veces en la solución del problema: Por lo que el programa debía detectar que durante esa partida el usuario había fallado cuatro veces antes de enviar el correo. Hubo que probar que el envío se hacía correctamente aunque se hubiera guardado la partida y se hubiera cargado posteriormente.
   b) Envío de pista al cabo de 30 minutos: Como en el caso anterior, el programa debía saber el tiempo transcurrido en la escena, para enviar o no el correo.

Una vez realizadas las pruebas de unidad, llevadas a cabo en este caso por el proyectando, se comenzó a realizar las pruebas de integración. La prueba de integración es una técnica sistemática para construir la arquitectura del software, mientras, al mismo

tiempo, se aplican pruebas para descubrir errores asociados en la interfaz. En nuestro caso, las pruebas de integración comprobaron, por ejemplo, el hecho de que si un objeto se cogía en una escena, se mantuviera en el inventario en la escena siguiente, o por el contrario, si un objeto usado en una escena y desaparecía del inventario, había que asegurarse de que no fuera visible en escenas posteriores. Otro ejemplo de prueba de integración fue la comprobación de que la transición de una escena a otra se hacía correctamente.

Las pruebas de validación se realizaron una vez que concluyeron las de integración. La validación se alcanza cuando el software funciona de tal manera que satisface las expectativas del cliente. [38]

Las pruebas de usabilidad consisten en seleccionar a un grupo de usuarios de una aplicación y solicitarles que la prueben.

3.4.1. Medidas de Usabilidad

Los usuarios elegidos para este trabajo han sido usuarios que nunca habían tenido excesivo contacto con el mundo de los problemas de lógica. Se eligieron así para poder apreciar el avance en sus habilidades en la resolución de estos problemas, ya que si se elegían usuarios familiarizados con este tipo de problemas, no se aportaría información valiosa.

A la hora de probar el juego con estos usuarios se tuvieron en cuenta los siguientes factores a medir:

a) Exactitud

Número de errores cometidos por los sujetos de prueba. Se observó los errores al usar el programa, como por ejemplo, a la hora de usar dos objetos, ya que los usuarios querían usarlos sin haberlos cogido.

Por otro lado, hablar con personajes del juego fue un punto que hubo que estudiar con cuidado. Es una parte crítica, ya que los personajes darán información para continuar avanzando en el juego y si la forma de hablar con ellos no es la apropiada, el jugador puede sentirse como si estuviera en un callejón sin salida, podría cansarse y abandonar el juego. Por tanto, la forma de mostrar los diálogos tuvo que ser lo más intuitiva posible, pulsando el botón “Hablar”, apareciendo las opciones a decir, y mostrar en la pantalla la pregunta hecha al personaje y su posterior respuesta.

También hubo que tener en cuenta cuántos errores tenía que cometer el usuario a la hora de resolver un mini-juego antes de enviar una pista. Vimos que la primera solución que daba el usuario en la resolución del juego, en la mayoría de las ocasiones, se realizaba al azar, en el segundo intento se comenzó a plantear el problema pero seguía

fallando. En la mayoría de las ocasiones se resolvía el problema en el tercer o cuarto intento, por tanto, se pensó que la mejor opción era enviar una pista si el usuario al cuarto intento, no había resuelto la prueba.

Al ser una interfaz sencilla, no hubo demasiados problemas de usabilidad respecto al número de errores del usuario en una prueba.

b) Tiempo requerido para concluir una actividad

En este trabajo era muy necesario conocer el tiempo requerido para ciertas actividades. En primer lugar se midió el tiempo de carga de una partida. A pesar de que no es tiempo empleado por el usuario en resolver un problema, es muy importante que la partida se cargue rápidamente con el fin de no aburrir al jugador. El tiempo de carga es bastante rápido y no supone ningún problema para los usuarios.

Otra actividad a medir fue el tiempo que tardaría un usuario en resolver un mini-juego. Si se cumplía el tiempo y no se había resuelto, se decidió enviar una pista por mail. Por lo que esta prueba ayudó a comprobar si el tiempo establecido inicialmente era adecuado.

c) Respuesta emocional

La respuesta emocional refleja cómo se siente el usuario al terminar la tarea. Si se siente satisfecho con lo aprendido al finalizar el videojuego. Los usuarios están satisfechos con el juego, con el modo de recibir las pistas y con el nivel de las pruebas lógicas. Aquellos que nunca habían jugado a un videojuego similar les resultó un poco más complicado manejar la interfaz, pero lo lograron sin problemas con ayuda del manual de usuario.

3.5. Problemas y decisiones tomadas

Ya que surgieron problemas previos al diseño del trabajo y ya que a lo largo del proyecto se han ido realizando modificaciones para que el programa no solo funcionara, sino que las decisiones tomadas fueran la opción óptima, a continuación se describen los problemas surgidos.

a) Diseño del trabajo a realizar

Este trabajo ha surgido de una idea del proyectando y se fue perfeccionando con la ayuda del tutor del trabajo. Se ha creado un trabajo desde cero, y por tanto, ha sido necesario un trabajo de investigación previo, el diseño de un guión para la historia a narrar y buscar las bases consistentes en las que se debía basar.

Para diseñar un buen juego, con un buen argumento, fue necesario consultar libros de aventuras, investigación y juegos. Por otro lado también fue

necesario conocer los dos tipos de juego que se iban a fusionar para dar lugar al estilo utilizado en este trabajo. Para conocer en profundidad los ARGs se jugó al famoso juego “In memorian II”, ya que a partir de él y de investigaciones previas al diseño del trabajo, se conocieron las características de este tipo de juegos, ya que éstos no eran tan conocidos como las aventuras gráficas, al menos para el proyectando.

b) Hablar con los personajes

Hablar con los personajes, o más concretamente la forma con la que se mostraban los diálogos con éstos. Se estableció desde un principio que ninguno de los personajes del juego hablaría de primeras con el protagonista, era éste el que tenía que comenzar los diálogos, eligiendo el personaje y pulsando el botón “Hablar”. Una vez pulsado, aparecían en un *JTextArea* las opciones de diálogo correspondientes, y se activaba el botón “Elige qué decir” para que se obtuviera la ventana donde se elegía la entrada de diálogo elegida. Siendo esto un poco complicado y poco intuitivo para el jugador, el cliente pidió una simplificación a la hora de hablar, por lo que el botón “Elige qué decir” desapareció y en su lugar, la ventana de opciones a la hora de hablar saltaba en cuanto se pulsaba el botón “Hablar”.

Otro problema que surgió fue cómo diseñar el almacenamiento de los diálogos de los personajes en el documento XML. Se decidió que la primera persona a hablar en la escena fuera Saúl, y como en toda aventura gráfica, debían aparecer varias opciones de diálogo. En un principio cada frase se consideraba un diálogo, pero no era una buena opción. Se pensó que lo mejor es que un diálogo estuviera formado por varias entradas de diálogo, siendo éstas todas las opciones que el usuario pudiera elegir a la hora de hablar. De esta manera, según el destinatario y la entrada de diálogo elegida, la respuesta era diferente. El atributo “dialogo” relaciona la respuesta con la entrada elegida. El atributo anterior sirve para obtener el diálogo anterior a la conversación actual, que en este ejemplo no existe al ser el primer diálogo con el personaje.

```
<DIALOGO id="1" DESTINATARIO="Chico" anterior="">

    <ENTRADA id="1" dialogo="5"  > 1: ¡Ey tío!, Menuda fiesta...</ENTRADA>

    <ENTRADA id="2" dialogo="6"  > 2: ¿Sabes que ha pasado?</ENTRADA>

    <ENTRADA id="3" dialogo="" > 3: Mmm, mejor nada... </ENTRADA>

</DIALOGO>
```

c) Narrador del videojuego

Un aspecto bastante complicado fue hacer diferenciar quien era el narrador del juego. Al realizar una aplicación que es una mezcla de varios estilos surge el

problema de cómo hacer para que tratándose de tipos de juego tan diferentes se fusionen y funcionen como uno. ¿Cómo hacemos para fusionar las aventuras gráficas y los ARGs? La respuesta es: con dos personajes protagonistas. El protagonista de la aventura conversacional-gráfica, Saúl, es el protagonista virtual, el que se moverá por todas las escenas y pedirá ayuda al protagonista de carne y hueso, que está sentado delante de su ordenador. Saúl va enviando emails de todo lo que ve y encuentra, sirviendo esta información como pista para el jugador del videojuego y pidiendo ayuda cuando encuentra trabas en el camino.

d) Creación de los mini-juegos

Al ejecutarse una partida el usuario puede ver la ventana principal del programa, donde están los elementos básicos de la sala en la que el jugador se encuentra, como sus objetos, los personajes, lo que lleva el jugador en el inventario, entre otros elementos. Cuando el usuario pulsa el botón "Usar" para poder usar dos objetos se pueden obtener dos consecuencias. Una consiste en la aparición de un nuevo objeto en la sala, que posteriormente el usuario tendrá que hacer alguna acción con él, y otra, la aparición de un mini-juego. Como se ha mencionado anteriormente, los mini-juegos son pruebas lógicas que el jugador debe resolver para avanzar en la partida y poder resolver el caso de secuestro de Lidia.

Para saber qué mini-juego se corresponde con la escena en la que se encuentra el usuario, y para que sea de forma general, se creó un elemento llamado "PRUEBA" dentro de cada elemento ESCENA del documento XML que refleja el estado de la partida. Cuando el software detecta que ha llegado el momento de crear un mini-juego, surgió el problema de cómo había que crear la instancia del mini-juego. En un principio, y tal como se estudió en las asignaturas de Java durante la carrera, se recurrió a la creación de la instancia del mini-juego mediante el nombre de la clase correspondiente al mini-juego y "new". Esta idea permitía ejecutar el programa, pero no era general, lo cual, es imprescindible en el mundo de la programación. No era una buena idea crear "a mano" una instancia del mini-juego, ya que si la prueba cambiaba de nombre, o se añadía o eliminaba alguna en el documento XML, conllevaría a hacer una modificación manual en un punto concreto del código. Para solucionar este problema se estudió la conocida "Reflexión en Java". El "API Reflection" es una herramienta muy poderosa, que a pesar de su potencia, es un API poco conocido, sobre todo para los principiantes en Java. [40]

e) Mostrar los avances en las partidas

Durante la ejecución del videojuego, el jugador tendrá que tomar decisiones, probar si su idea para progresar es correcta, o sin embargo, si tiene que seguir pensando. Las acciones que no conducen a la solución mostrarán un mensaje que va cambiando de forma aleatoria, para dar credibilidad al personaje del juego y no cansar al usuario, en la pantalla de texto comunicando que esa acción no puede realizarse. Por el contrario, cuando la acción realizada, como por ejemplo,

coger un objeto de la sala, hablar con un personaje o usar dos objetos, es correcta, es necesario reflejar las consecuencias de la realización de dichas acciones. En un principio no se sabía como “decir al programa” lo que debía hacer. Se pensaba que el documento XML solo era necesario para almacenar elementos de la escena como objetos y sus personajes. Pero el documento XML debía guardar no solo mucha más información que la que se pensaba inicialmente, sino toda la información correspondiente a la partida. Para ello, se decidió poner elementos “CONSECUENCIA” en el documento XML con el fin de que cuando el software detectara que se había cumplido la condición correspondiente, se cargaran las consecuencias oportunas al estado de la partida.

A la hora de la ejecución, podrían darse ciertas casuísticas que darían lugar a varias consecuencias diferentes, como por ejemplo:

- **Consecuencia “Mirar un objeto”**

Mirar un objeto, en ciertas ocasiones, permite la carga de un mini-juego. Una forma bastante clara de reflejar en el documento XML que es necesario cargar la prueba, es añadir un elemento llamado “CONSECUENCIAMIRAR” como hijo del elemento “OBJETO” mirado. Este elemento “CONSECUENCIAMIRAR” debe reflejar la acción a realizar al mirar este objeto, es decir, cargar un mini-juego.

```
<CONSECUENCIAMIRAR>

    <PRUEBA>AdivinaPinturaPared2.class</PRUEBA>

</CONSECUENCIAMIRAR>
```

Al mirar el objeto correspondiente, la consecuencia sería la aparición del mini-juego llamado “AdivinaPinturaPared2”.

- **Consecuencia al resolver una prueba lógica**

Cuando un mini-juego se ha resuelto correctamente, el programa acude al documento XML con el fin de conocer la consecuencia. Una posible consecuencia sería la visibilidad de un objeto perteneciente a dicha escena, que anteriormente no se podía ver y por tanto, el jugador no tenía ni idea de que existía tal objeto. A partir de entonces, dicho objeto puede verse y se puede obtenerse su descripción y, dependiendo del objeto que sea, cogerse y usarse con otros objetos que el usuario tenga en el inventario o con los objetos propios de la sala.

```
<CONSECUENCIAPRUEBARESUELTA>

    <OBJETO VER="si">Mi casa</OBJETO>

</CONSECUENCIAPRUEBARESUELTA>
```

- **Consecuencia “Hablar con”**

Al hablar con personajes del juego, el jugador obtiene información necesaria para avanzar en la investigación. Una consecuencia posible de hablar con un personaje es “conocer” que dos objetos pueden usarse. En este caso se reflejó de la siguiente forma que al hablar con el personaje correcto la “Caja fuerte” habría de usarse con la “llave”. Al usuario no se le dice explícitamente que pueden usarse estos dos objetos, sino que anteriormente si el jugador usaba dos de estos objetos, no se le permitía realizar la acción ya que no existía ninguna conexión lógica. Al obtener la información proporcionada al hablar, esta acción ya sí es posible.

<CONSECUENCIAHABLAR USARCON="Llave">Caja fuerte

</CONSECUENCIAHABLAR>

- **Consecuencia “Usar con”**

Al usar un objeto con otro existen dos consecuencias posibles:

La primera que vamos a mencionar es la aparición de un objeto o varios, en la escena correspondiente. En el siguiente ejemplo se refleja que al usar la “Caja fuerte”, con el objeto indicado, la consecuencia será la visibilidad (VER=”si”) del objeto “Visa” y del objeto “Secuencia de números”.

<OBJETO VER="si" VISTO="no" DESCRIPCION=" Es una caja fuerte donde mi padre esconde la visa lejos de mi madre."

USADO="no" COGER="no" COGIDO="no" USARCON="">Caja fuerte

<IMAGEN>./resources/CajaFuerte.jpg</IMAGEN>

<CONSECUENCIAUSAR>

<OBJETO VER="si">Visa </OBJETO>

<OBJETO VER="si">Secuencia de números</OBJETO>

</CONSECUENCIAUSAR>

<CF VALOR="si">VISTO</CF>

</OBJETO>

La segunda consecuencia es que aparezca un mini-juego a resolver.

<OBJETO VER="si" VISTO="no" DESCRIPCION="Son 3 jeringas de 8ml, 5ml y 3ml."

COGER="si" COGIDO="no" USARCON="Dormican">Pack de jeringas

<IMAGEN>./resources/Pack de jeringas.jpg</IMAGEN>

<CONSECUENCIAUSAR>

<PRUEBA>DuermeAlPerro2.class</PRUEBA>

</CONSECUENCIAUSAR>

<CF VALOR="si">VISTO</CF></OBJETO>

f) Detectar el fin de escena y pasar a una nueva

En un principio no se sabía cómo realizar los cambios de pantalla. Estaba claro que se tenían que cumplir ciertas condiciones necesarias para poder pasar al siguiente escenario. Para poderlo llevar a cabo, se creó el elemento "CF", es decir, "Condición Fin" que refleja el estado que debe tener cierto atributo de un objeto para poder dar la condición como cumplida. En una misma escena habrá varias "condiciones de fin de escena" y solo podrá pasar de una escena a otra cuando se cumplan todas. En el siguiente ejemplo puede verse que la condición fin de escena es que la prueba "Círculo" cambie su atributo a "RESUELTA=si".

<PRUEBA RESUELTA="no" VER="si" ENVIADA="no" INTENTOS="0">Circulo

<PATH>./Circulo.class</PATH>

<CONSECUENCIAPRUEBARESUELTA>

<OBJETO VER="si">Policía</OBJETO>

</CONSECUENCIAPRUEBARESUELTA>

<CF VALOR="si">RESUELTA</CF>

</PRUEBA>

g) Desaparición de los objetos usados

En toda aventura conversacional y/o gráfica existe un inventario donde se van guardando los objetos que el jugador va encontrando en su camino. La correcta utilización de dichos objetos sirve para desencadenar consecuencias y avances en la partida.

El inconveniente es que en este tipo de juegos se acumula una gran cantidad de objetos en el inventario. El jugador encuentra un objeto en una escena y puede utilizarlo con otro en escenas bastante posteriores. Cuando dos objetos se usan, deben desaparecer del inventario, ya que llevarían a una confusión al jugador, es decir, es posible que el jugador quisiera volverlos a usar con otros. Por ello, se decidió que la mejor elección sería hacerlos desaparecer una vez que se usaron. En un principio, los objetos no iban

desapareciendo, pero se llegó a esta conclusión después de tener una entrevista con el tutor.

Para hacerlos desaparecer se recurrió de nuevo al documento XML. En él, se creó un elemento denominado “INVENTARIO” con el fin de guardar los elementos hijos que fueran los objetos cogidos durante la partida. Dichos objetos tendrían que tener un atributo que aportara información al estado del objeto. Por ello se creó el atributo “USADO”. Si el objeto había sido usado, dicho atributo tendría el valor “SI” y cuando se repintara la pantalla principal, no aparecería como elemento del inventario.

La desaparición de los objetos se complica cuando hay que encontrar el momento correcto para hacer este cambio en el atributo “USADO”, ya que si no se realiza correctamente y se produce la desaparición temprana de dichos objetos, el jugador no podría volverlos a usar.

Lo primero que había que dejar claro era el tipo de consecuencia al usar dichos objetos. Una vez resuelto este problema, la cuestión era saber cuándo modificar el atributo de los objetos. Inicialmente, estos objetos tenían el atributo “USADO” =”NO”.

Para que el programa principal detectara que el mini-juego se había resuelto, se tuvo que usar la anteriormente mencionada “Reflexión de Java”. Una vez creado el objeto de la clase correspondiente al mini-juego, éste era el encargado de comunicar a la pantalla principal si la prueba se había resuelto correctamente. Se usó la funcionalidad de la reflexión ya que permitiría la conexión entre el objeto del mini-juego, la pantalla principal y el objeto de la clase lector que guardaría las modificaciones en el árbol JDOM que posteriormente crearía el documento XML. Una vez establecida dicha unión, la modificación de los atributos era simple, poner “USADO=SI” si la prueba se había resuelto correctamente y “USADO=NO” en caso contrario.

h) Salir de los mini-juegos

Un menor problema fue la forma de cerrar el mini-juego. Se producía error cuando se cargaba la prueba lógica ya que una vez usados los objetos, si el jugador no quería resolverla en ese momento, y por tanto, pulsaba la “X” para salir del mini-juego, los objetos habían desaparecido del inventario. Sin embargo, funcionaba perfectamente si se pulsaba el botón “Cerrar” de la propia pantalla de la prueba. Para ello se utilizó el método *setDefaultCloseOperation* con la opción *DISPOSE_ON_CLOSE* para que la pantalla principal quedara en las mismas condiciones que cuando el mini-juego fue ejecutado y así no se cerraran la ventana principal y la ventana de la prueba.

i) Envío de las pistas al jugador

Para lograr que el juego fuera realista, se pensó que una buena manera de lograrlo era haciendo que el jugador recibiera emails de Saúl. Para ello se tuvo que acudir a la biblioteca JavaMail para conocer los métodos necesarios para la recepción de las pistas. Para poder usar esta biblioteca era necesaria la elección de un protocolo para la transferencia del correo electrónico. Se usó el protocolo SMTP que permite la transferencia de correo de un servidor a otro mediante una conexión punto a punto. Es un protocolo que funciona en línea, encapsulado en una trama TCP/IP. El correo se envía directamente al servidor de correo del destinatario.

Uno de los problemas que surgieron fue el diseño de la recepción de las pistas, las cuales se reciben por dos motivos. Uno de ellos es si el jugador introduce mal la solución de la prueba en cuatro intentos. Otro motivo por la que el usuario recibe un email es si pasa un tiempo determinado sin solucionar el mini-juego. El problema que se plantea es si el jugador guarda la partida justo antes de recibir el email. Si en esta situación el jugador carga la partida, se tendría que volver a esperar de nuevo el tiempo establecido para la recepción de la pista. Para evitar esta situación se pensó en guardar el tiempo trascurrido en un elemento del documento XML cuando se guarda la partida, así cuando se vuelve a cargar la escena correspondiente, el tiempo cuenta desde ese tiempo guardado y no desde cero, por lo que sí se estuvo a punto de recibir el mail al guardar, lo recibirá en cuanto vuelva a cargar la partida.

j) Medida del tiempo para enviar una pista

Para medir el tiempo en el que había que enviar una pista al jugador se usó primeramente un bucle que midiera el tiempo transcurrido calculando la diferencia entre el tiempo actual (en milisegundos) y el tiempo guardado en la partida. Al realizar este bucle, si el tiempo en el que había que enviar la pista no era demasiado pequeño, la carga de una partida entraba en el bucle y no se cargaba correctamente. Finalmente para evitar este error se utilizó finalmente un objeto *Timer*, asociado a un escuchador, para avisar cuando se había cumplido el tiempo establecido y enviar la pista en el instante adecuado. Conocer el tiempo en el que se enviaba la pista era complejo, ya que el tiempo cambiaba según lo que hubiera jugado el usuario, es decir, no siempre se enviaba la pista a los 30 minutos, se enviaba a los 30 minutos de tiempo jugado. Si el usuario había jugado 15 minutos y había decidido guardar la partida, cuando la iniciase, no se enviaría a los 30, si no a los 15 minutos restantes.

k) Codificación JDOM

Todo documento JDOM necesita una especificación de la codificación a usar. Por defecto la codificación es UTF-8. Con esta codificación se tenían muchas limitaciones ya que no se podían representar caracteres como las tildes o la letra

"ñ". Para solucionar este problema, se eligió la codificación ISO-8859-1, ya que permite usar símbolos latinos.

Al guardar las partidas JDOM usaba la codificación UTF-8. Por tanto, hubo que cambiar las opciones de JDOM para que guardase los documentos XML con la codificación ISO-8859-1 y así no tener caracteres inválidos.

l) Introducción de sonido y vídeos

Los elementos multimedia despiertan la atención y el interés del usuario en el juego, lo hacen más atractivo. Para introducir dichos elementos hubo que hacer uso de la biblioteca JMF.

Uno de los problemas a solucionar a la hora de introducir el vídeo, fue la reproducción del vídeo y el audio. Cuando se reproducía el video, el sonido no se podía escuchar. A pesar de que JMF admite varias extensiones de reproducción de vídeos, el sonido no se escuchaba si no se usaba mpg, incluso hubo que elegir el correcto número de *frames/seg*.

Otro problema a resolver fue cómo hacer que el programa detectara el fin del vídeo que sirve de introducción al juego. Cuando este vídeo finalizara, el programa principal tenía que mostrar la ventana principal donde el jugador debía empezar a jugar. En un principio se añadió en el documento XML un elemento llamado "DURACION" y que tuviera el valor de la duración del vídeo. Esta solución no era óptima, ya que al cambiar el vídeo, también era necesario cambiar el valor de este elemento. Para ello, se pensó que lo óptimo sería que el programa detectara el fin de la reproducción mediante un evento y así, el programa detectaría cualquier vídeo de introducción al juego.

4. CONCLUSIONES Y TRABAJOS FUTUROS

4.1. Enriquecimiento personal

4.2. Posibles usuarios

4.3. Trabajos y mejoras futuras

4.4. Conclusiones

4.1. Enriquecimiento personal

El desarrollo de este videojuego le ha aportado al autor ciertas habilidades, que hasta que no se visto forzado a ejercitarlas, no ha sabido cuál iba a ser el resultado. Se citan a continuación:

a) Desarrollo de la imaginación. Para poder escribir una historia, con un guión de diálogos incluido, se necesita tener cierta imaginación y cierta adquisición de ideas, ya que la escritura de un guión no es una actividad ni mucho menos trivial. Esta actividad es más difícil de lo que parece, ya que se buscaba crear un misterio que captara la atención del jugador.

b) Adquisición de conocimientos sobre XML: cómo diseñar un documento XML que sirva como base de datos de una partida del videojuego.

c) Adquisición de conocimientos sobre el uso de librerías externas (que no se encuentran por defecto en el JDK) de Java: cómo poder utilizar ciertas librerías como JDOM, JavaMail, JMF, etc., siendo librerías externas a Java. Consecuentemente, el conocimiento de los métodos de dichas librerías para poder usarlas de forma correcta.

d) Utilidad de conocimientos adquiridos en la carrera: cómo programar el videojuego utilizando los conocimientos aprendidos de Java, una estructura de clases, métodos, objetos, eventos, etc.

e) Mayor conocimiento sobre la materia: ver el grado de dificultad que tiene el desarrollo de un videojuego, etapas de creación, motor de juego, aplicaciones de desarrollo de videojuegos, etc., papel actual de éstos en la educación.

f) Conocimiento de UML: diagramas de casos de uso, diagrama de secuencias, diagramas de estado y diagramas de clase que permitieran un buen modelado.

4.2. Posibles usuarios

Este videojuego puede ser jugado por cualquier persona, sepa o no de programación, quiera o no aprender los principios de la algoritmia, ya que se considera que aún así va a pasar un rato agradable y entretenido, siendo principalmente dirigido a:

a) Estudiantes universitarios: Principalmente en los primeros años de carrera, que es el momento en el que necesitan desarrollar habilidades necesarias para el posterior estudio de la algoritmia.

b) Estudiantes de un módulo de este tipo de programación: Para aquellos estudiantes de FP (Formación Profesional), que se vayan a iniciarse en el mundo de la algoritmia y programación.

4.3. Trabajos y mejoras futuras

En este apartado se van a comentar aquellos trabajos o mejoras futuras que se pueden realizar, y que como continuación a este proyecto, otra persona podría continuar:

- ✓ Convertir el videojuego en un juego multijugador, los denominados MMOG ("Massively Multiplayer Online Game"), que actualmente están muy de moda. En dicho juego se crearían foros y chats en los que los usuarios podrían compartir su información y opinión sobre la investigación.
- ✓ Adición de más vídeos al juego, para aumentar el realismo de la historia.
- ✓ Crear una web con información acerca de la investigación donde se puedan aportar más pistas, además de las enviadas al correo electrónico.
- ✓ Añadir una ventana de selección de jugador. Para ello se le puede dar al jugador a elegir en una ventana el personaje que desee, y añadir la foto de ese personaje en la pantalla principal, por ejemplo, sustituyendo la imagen seleccionada por uno de los logotipos existentes en la pantalla principal.
- ✓ Cambiar los diálogos en función de si el jugador ha hablado ya o no con un determinado personaje. Esto es algo muy habitual en los juegos comerciales, ya que la información que proporciona un personaje no siempre es la misma y en función de los objetos que se hayan conseguido y las acciones que se hayan realizado, la información que aporta es diferente.
- ✓ Añadir mini-juegos más complejos, que utilicen más técnicas de resolución de problemas y otras pruebas que impliquen algoritmos famosos, como de ordenación, búsqueda, etc. Para ello crear una clase padre de la que hereden los métodos comunes, de esta forma la implementación será mucho más cómoda y se podrán añadir más variedad de juegos.

4.4. Conclusiones

Basándose en los estudios realizados y mencionados a lo largo de este trabajo, a partir de los cuales se puede demostrar la eficiencia de los juegos de lógica en el aprendizaje de los algoritmos y basándose en las nuevas necesidades de enseñanza debido al cambio generacional producido, entre otros motivos, por el avance tecnológico, se ha creado

este videojuego educativo que pretende capturar la atención del jugador desde el primer momento.

Para poder llegar al final de la investigación el jugador debe demostrar su avance en la resolución de los problemas de lógica ya que debe resolver los problemas por sí solo.

La investigación que el jugador debe hacer para resolver los problemas planteados es un ejercicio de auto-aprendizaje que servirá para desarrollar las habilidades necesarias para el aprendizaje y diseño de algoritmos, además de adquirir las ventajosas habilidades que se adquieren en este tipo de videojuegos.

El resultado final de este trabajo ha sido satisfactorio ya que aquellos usuarios que lo han probado han jugado hasta el final admitiendo un progreso de sus conocimientos en el ámbito de la resolución de problemas. Este trabajo ha sido también una experiencia positiva desde el punto de vista del desarrollador ya que se han conseguido los objetivos inicialmente planteados y superados los problemas surgidos a lo largo del trabajo.

ANEXO A: DIAGRAMAS DEL SOFTWARE.

1. ¿Por qué modelamos?

El modelado es una parte central de todas las actividades que conducen a la producción de buen software. Se construyen modelos para comunicar la estructura deseada y el comportamiento de nuestro sistema. Se construyen modelos para visualizar y controlar la arquitectura del sistema. Se construyen modelos para comprender mejor el sistema que se está realizando, muchas veces descubriendo oportunidades para la simplificación y la reutilización, es decir, se construyen modelos para controlar el riesgo. En definitiva, se modela para comprender mejor el sistema que estamos desarrollando. [3]

A través del modelado, conseguimos cuatro objetivos:

- ✓ Los modelos ayudan a visualizar cómo queremos que sea un sistema.
- ✓ Los modelos permiten especificar la estructura o el comportamiento de un sistema.
- ✓ Los modelos proporcionan plantillas que guían en la construcción de un sistema.
- ✓ Los modelos documentan las decisiones que se han adoptado.

2. Diagramas de secuencias

Un diagrama de interacción muestra una interacción, que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes.

2.1. Diagrama de secuencias de una escena nueva

Mediante este diagrama se aprecia la secuencia de pasos que se siguen al arrancar el juego. Tras crear la ventana inicial de opciones de partida, el usuario elige “Nueva Partida”, una de las opciones del menú principal, lee la información del juego, rellena los datos del objeto Formulario, se crean objetos de clases necesarias para mostrar la pantalla principal del juego. Cuando el jugador ha de resolver una prueba se crea el objeto final *Minijuego*. Una vez que el jugador haya resuelto el acertijo, el objeto *Minijuego* devolverá el resultado para que la pantalla principal se quede esperando a las siguientes acciones del jugador. La pantalla principal es capaz de detectar el momento en el que es necesario el envío de una pista ya que el jugador no sabe cómo seguir.

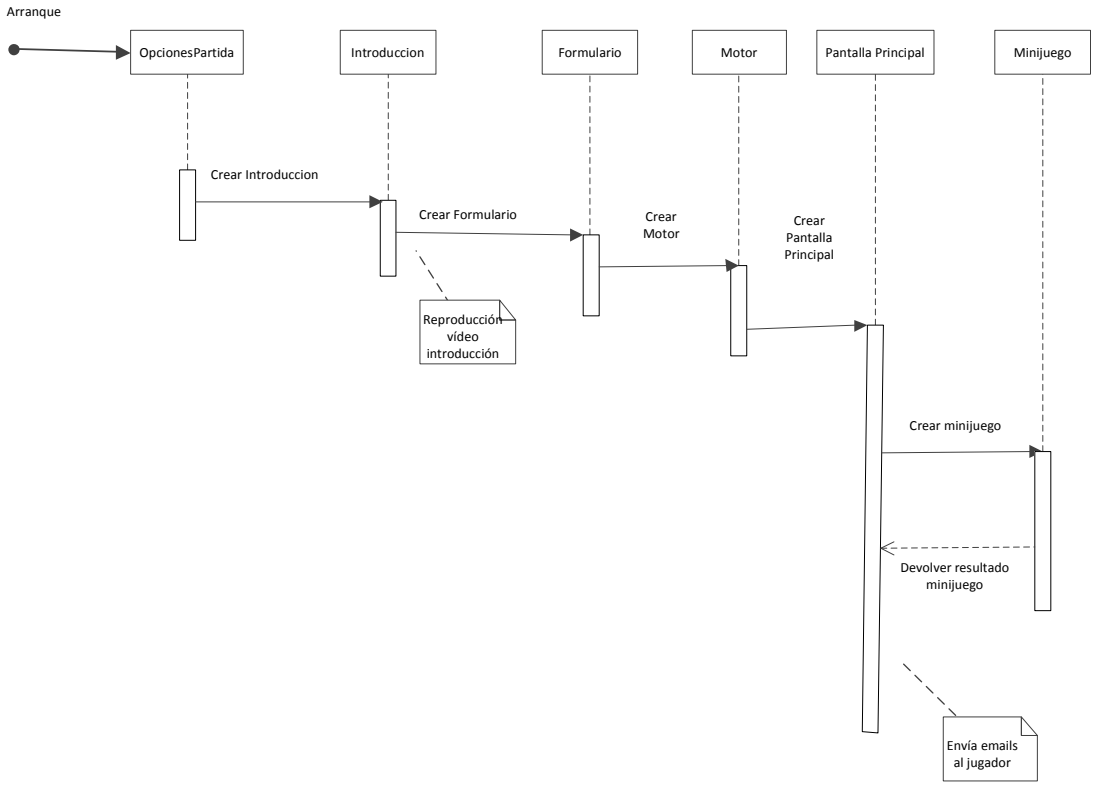

Ilustración 46 Diagrama de secuencias de una escena nueva

2.2. Diagrama de secuencias de una escena cargada

Mediante este diagrama se aprecia la secuencia de pasos que se siguen al arrancar el juego. Tras crear la ventana inicial de opciones de partida, el usuario elige una de las partidas guardadas, se crean objetos de clases hasta que se crea el objeto final *Minijuego*. Una vez que el jugador haya resuelto la prueba, el objeto *Minijuego* devolverá el resultado para que la pantalla principal se quede esperando a las siguientes acciones del jugador. La pantalla principal es capaz de detectar el momento en el que es necesario el envío de una pista ya que el jugador no sabe cómo seguir.

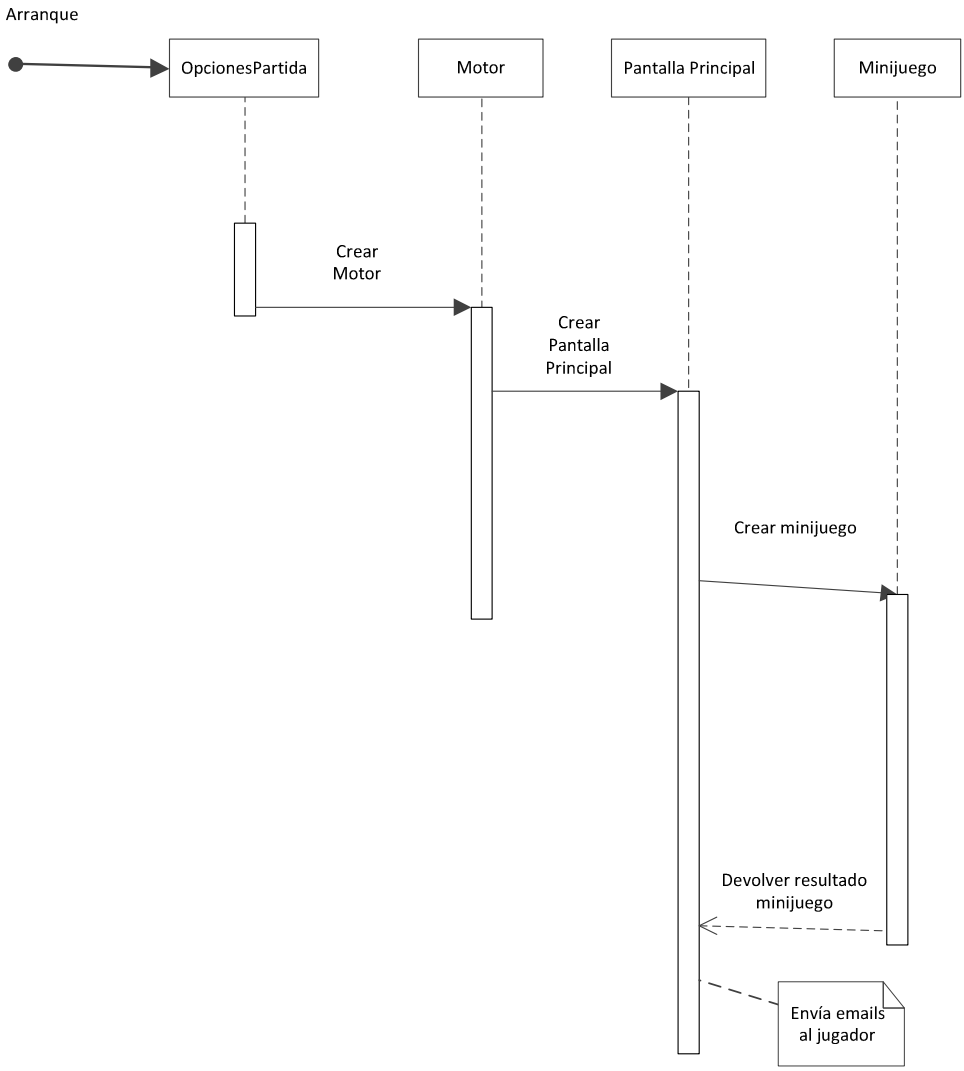

Ilustración 47 Diagrama de secuencias de carga de una escena

3. Diagrama de estados

Un diagrama de estados muestra una máquina de estados, destacando el flujo de control entre estados. Una máquina de estados es un comportamiento que especifica las secuencias de estados por las que pasa un objeto a lo largo de su vida en respuesta a los eventos, junto con sus respuestas a esos eventos. Un estado es una condición o situación en la vida de un objeto durante la cual satisface alguna condición, realiza alguna actividad o espera algún evento. El diagrama de estados utilizado para modelar los aspectos dinámicos del sistema es el que se muestra a continuación:

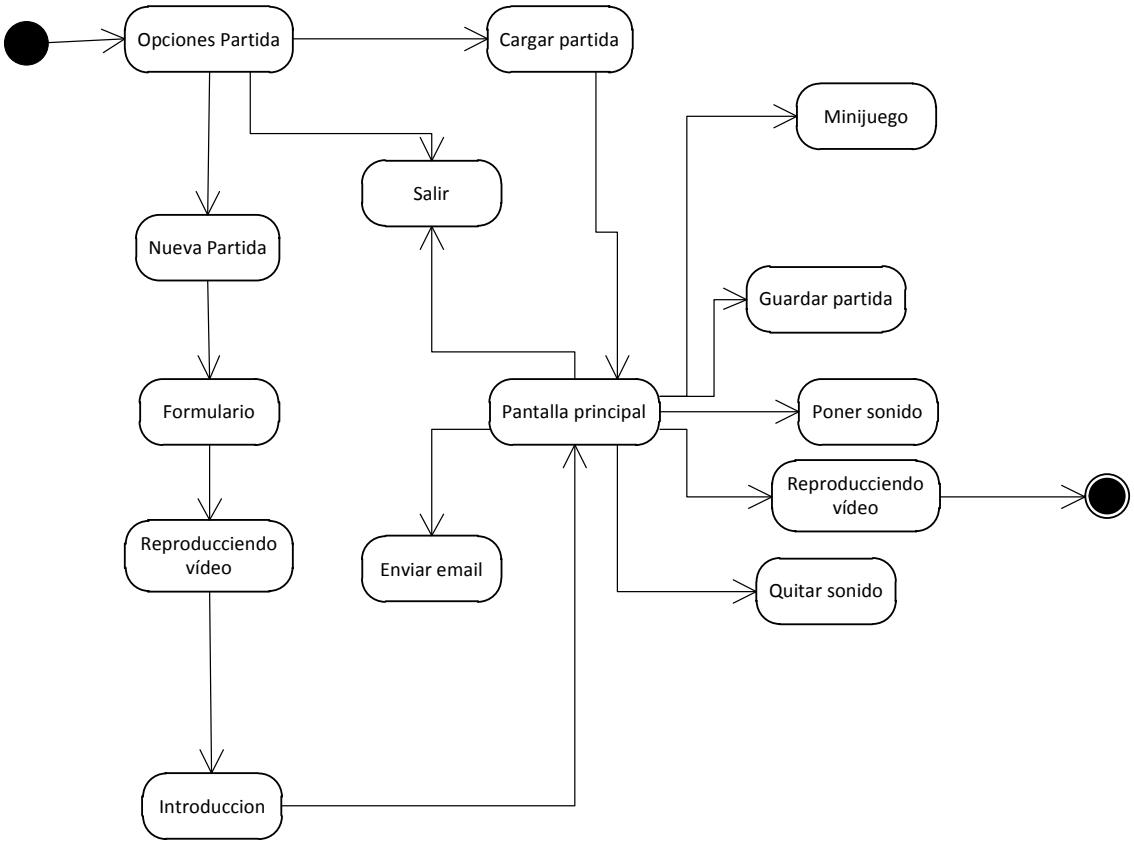

Ilustración 48 Diagrama de estados

4. Diagrama de clases

Un diagrama de clases muestra todas las clases que intervienen en un proyecto y las relaciones que existen entre ellas. Se utilizan para modelar la vista de diseño estática de un sistema. Esta vista soporta principalmente los requisitos funcionales de un sistema, los servicios que el sistema debe proporcionar a sus usuarios finales.

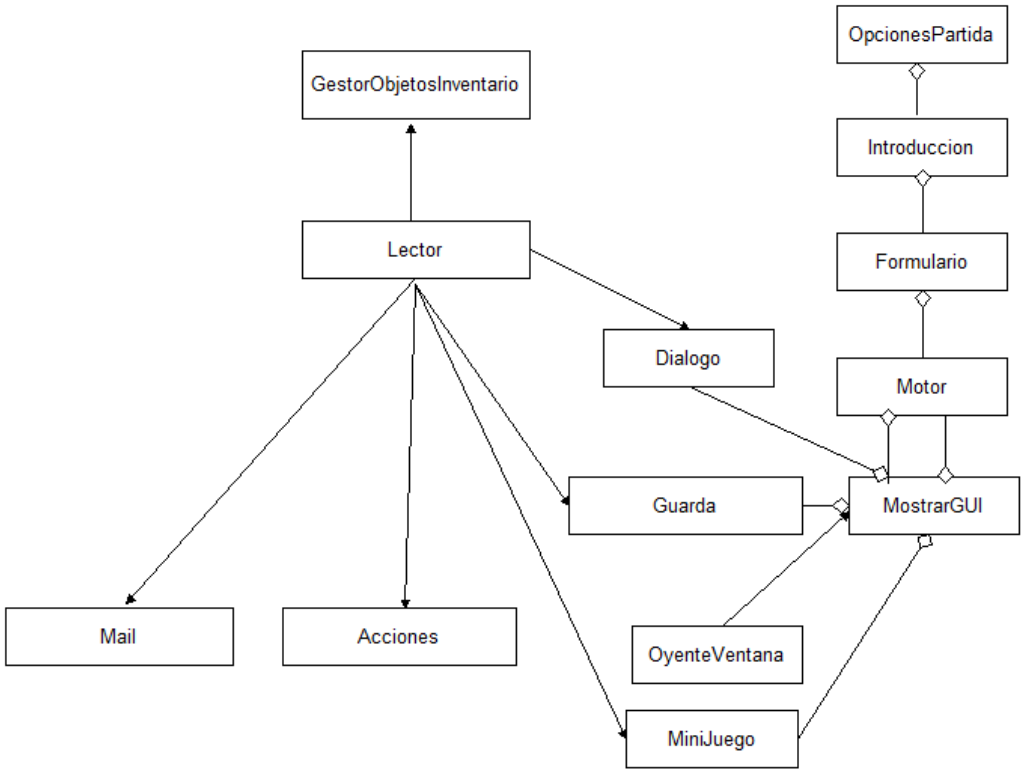

Ilustración 49 Diagrama de Clases

A continuación se muestran los diagramas UML de las clases del proyecto:

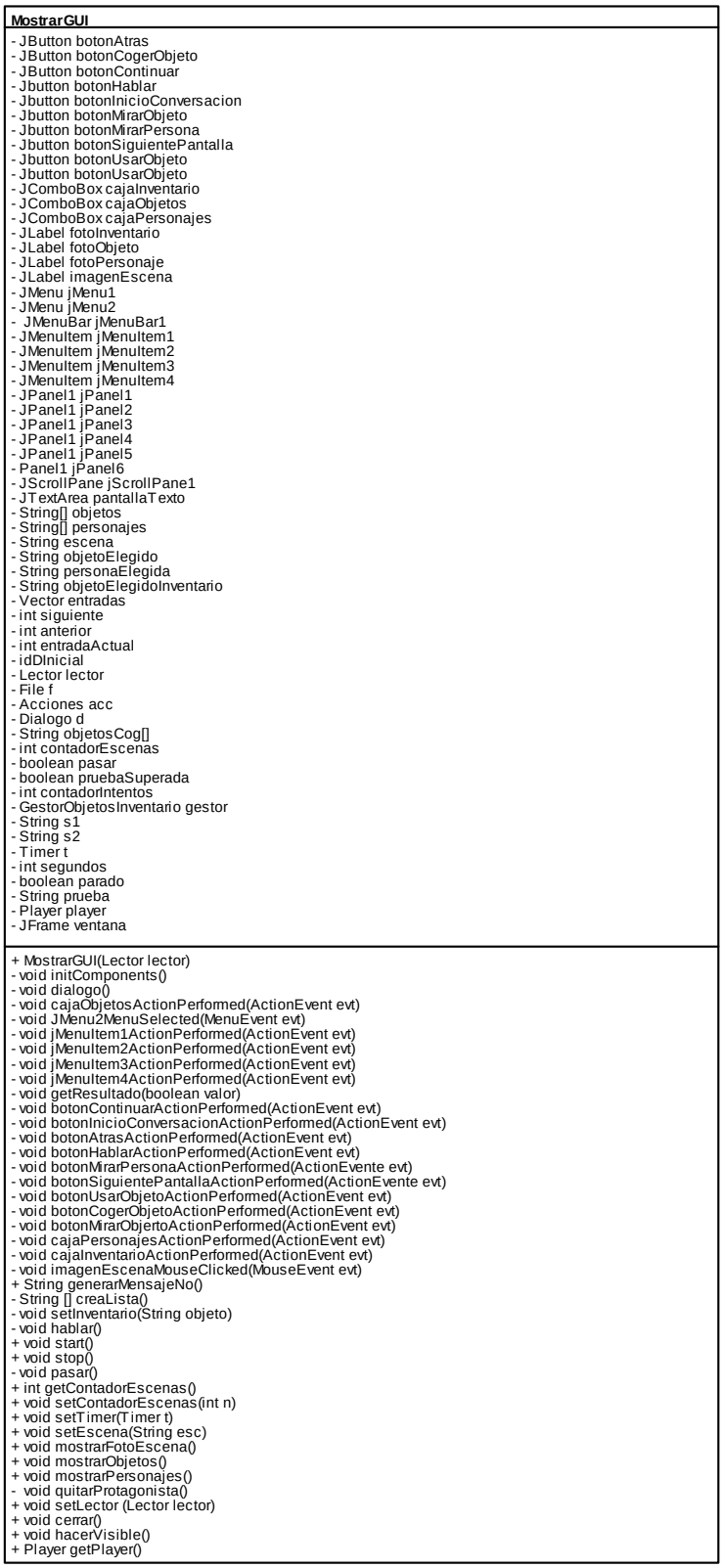

Ilustración 50 Clase MostrarGUI

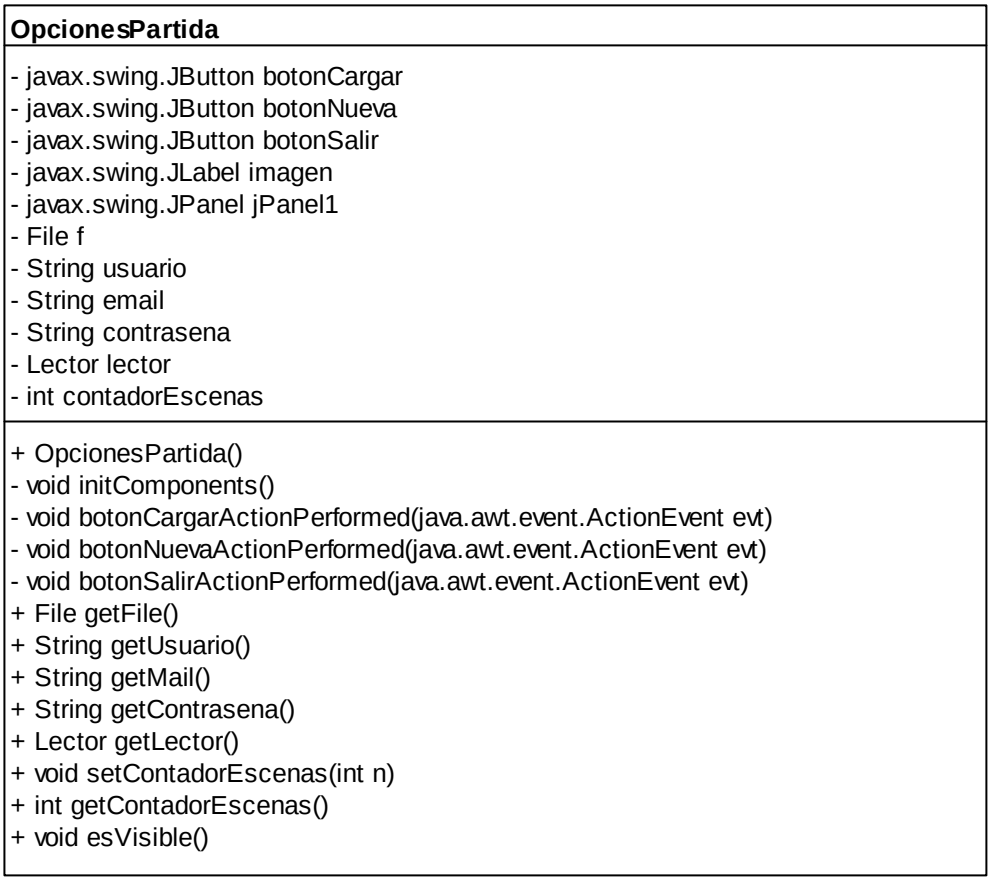

Ilustración 51 Clase OpcionesPartida

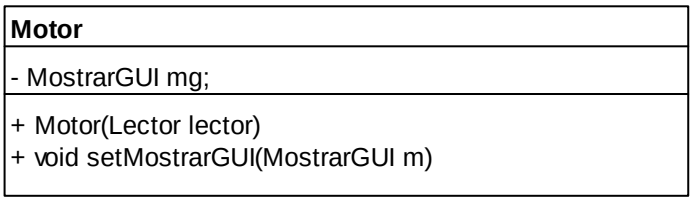

Ilustración 52 Clase Motor

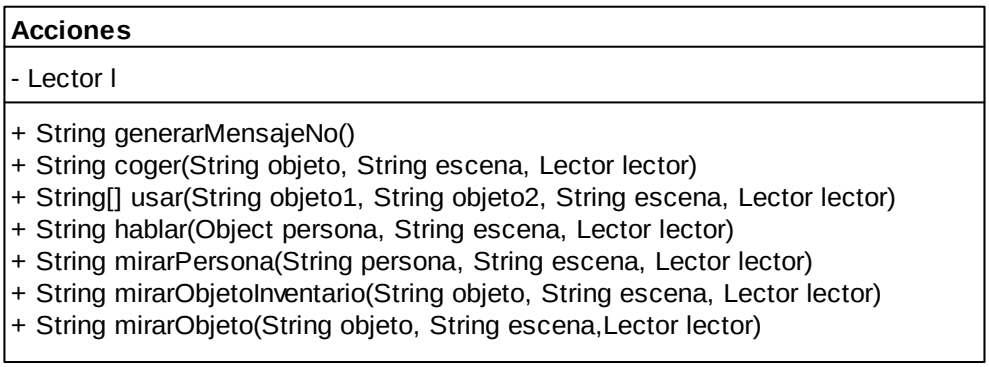

Ilustración 53 Clase Acciones

Formulario
- javax.swing.JButton botonDatos - javax.swing.JTextField campoEmail - javax.swing.JTextField campoNombre - javax.swing.JLabel foto - javax.swing.JLabel jLabel2 - javax.swing.JLabel jLabel3 - String nombre - String email - boolean todoCorrecto - Lector lector - boolean parado - Timer t - JFrame ventana - Player player
+ Formulario(Lector lector) - void initComponents() - void init() + void iniciarVideo() + void start() + void stop() - void botonDatosActionPerformed(java.awt.event.ActionEvent evt)

Ilustración 54 Clase Formulario

Guarda
+ void guardaXML(Document document, String partida) + File cargaXML(String partida) + String asociaPartidaConUsuario(String partida) + String[] listarDirectorio()

Ilustración 55 Clase Guarda

Introduccion
- Lector lector; - javax.swing.JScrollPane jScrollPane1 - javax.swing.JTextPane jTextPane1
+ Introduccion(Lector lector) - void initComponents() - void jTextPane1MouseClicked(java.awt.event.MouseEvent evt)

Ilustración 56 Clase Introducción

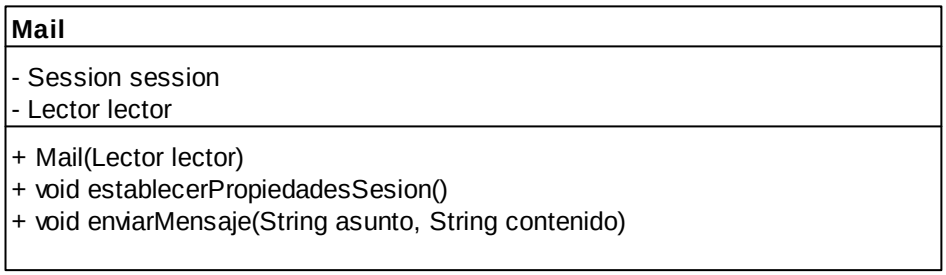

Ilustración 57 Clase Mail

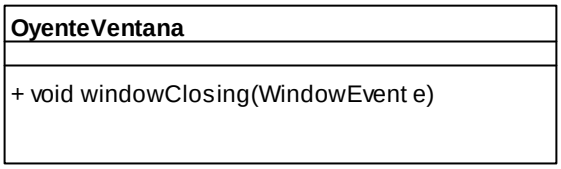

Ilustración 58 Clase OyenteVentana

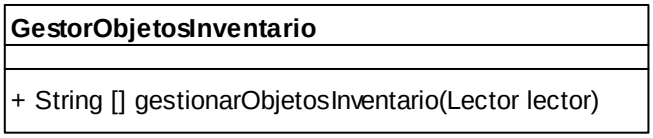

Ilustración 59 Clase GestorObjetosInventario

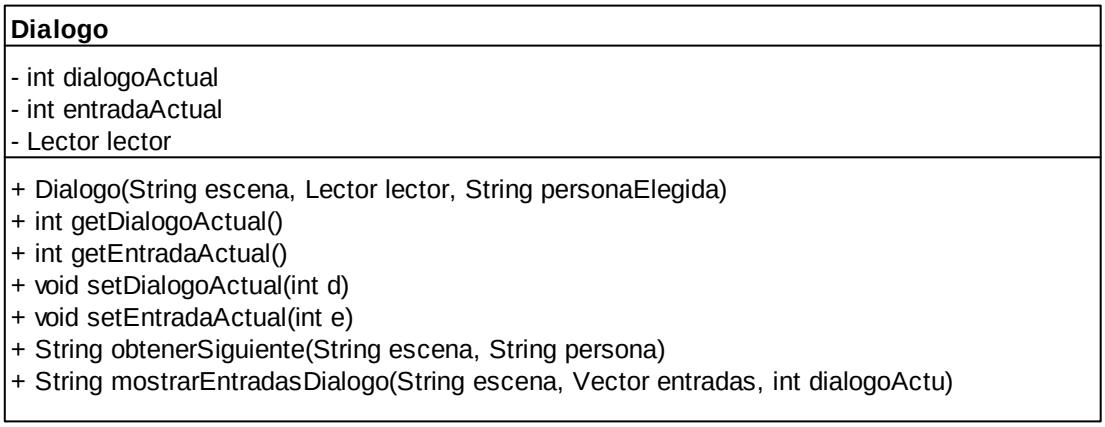

Ilustración 60 Clase Dialogo

Ilustración 61 Clase Lector

Vitrina2
- JButton botonCerrar - JButton botonMover - JTextField campoEntrada1 - JLabel cuarta - JLabel cuarta1 - JLabel cuarta2 - JLabel primera - JLabel primera1 - JLabel primera2 - JLabel segunda - JLabel segunda1 - JLabel segunda2 - JLabel tercera - JLabel tercera1 - JLabel tercera2 - JLabel platoMaya - JLabel jcontador - JTextArea instrucciones - JScrollPane jScrollPane1 - String vitrina1[] - String vitrinaAux[] - String vitrina2[] - int contador - String platoAMover - String origen - boolean pruebaSUperada - Lector lector - Motor m - MostrarGUI interfazGUI
+ Vitrina2() - void initComponents() + void comenzar() - void BotonCerrarActionPerformed(ActionEvent evt) + void comprobarMovimiento(String plato, int origen, int destino) + boolean mirarPlatosMenores(String plato, int destino, int posicion) + void mirarEncima(String plato, int origen, int posicion) + void moverDestino(String plato, int origen, int destino, int posicion) + int[] buscar(String plato) + boolean getPruebaSuperada() - void botonMoverActionPerformed (ActionEvent evt) + void getPruebaInt(Lector l, MostrarGUI mg) + boolean comprobarResultado() + void esVisible(Lector lector)

Ilustración 62 Clase Vitrina2

ANEXO B: MANUAL DE JUEGO

1. Instalación y comienzo de juego

Paso 1: Instalación

Para la instalación del videojuego es necesario seguir los pasos correspondientes que aparecen al hacer doble *click* sobre "Rescue.exe". El directorio donde se ha de instalar este juego debe ser C:\. Para ejecutar el archivo .zip descomprímelo y haz doble *click* sobre el archivo .bat. Para que el juego se ejecute correctamente se debe instalar en el ordenador el archivo *jmf-2_1_1e-windows-i586.*

Paso2: Empieza a jugar

En la ventana "Opciones de partida", haz clic en el botón "Nueva partida" si deseas jugar o "Cargar partida" si anteriormente ya has jugado y tenias una partida guardada.

2. Cómo se juega a Rescue

Verano de 2011

Es verano y en una zona de Madrid un grupo de amigos, entre los que se encuentra Lidia, deciden dar una fiesta. Todo transcurre con normalidad hasta que un grito interrumpe la diversión. Lidia ha desaparecido.

Equipo de búsqueda

En la fiesta está un chico llamado Saúl, que está coladito por Lidia. Pone en marcha un equipo de investigación contando con tu ayuda.

3. Pantalla de juego

A. Vista principal

Muestra en todo momento el lugar en el que se encuentra Saúl. Desde ella puedes dirigir el rumbo de la aventura: decidir lo que Saúl va a hacer, con quien deseas que dialogue y qué objetos va a examinar, coger o emplear.

B. Panel de opciones

En el panel de opciones tienes las siguientes acciones:

- Guardar partida. Si necesitas hacer un descanso en tu investigación tienes la posibilidad de guardar tus progresos.
- Cargar partida. Después de un descanso merecido puedes reanudar tu investigación en el punto en el que lo dejaste.
- Poner sonido. Si deseas un poco de música de fondo.
- Quitar sonido. Si te has cansado de escuchar la música de fondo no tienes porqué torturarte, la puedes quitar.
- Salir del juego. Si necesitas descansar de pensar puedes salir del juego. Si quieres almacenar el estado de la partida es necesario haberla guardado antes.

C. Inventario

En el inventario se almacenan todos los objetos que Saúl recoge durante el juego y de los que debe servirse para avanzar en la aventura.

4. **Empieza a jugar**
   A. Comienza la aventura

      Presta atención al vídeo que da comienzo a la aventura. Tras la introducción Saúl te pondrá al día de los hechos ocurridos. Lidia ha desaparecido y se han encontrado unas pinturas extrañas en las paredes de su habitación.

   B. Exploración

      Tu primer objetivo es descubrir qué son esas pinturas. Para ello, debes explorar los alrededores y hablar con los posibles testigos. Para ello puedes mirar/coger/usar objetos y mirar/hablar con los personajes de la escena.

   C. Conversación

      Saúl debe comenzar y dirigir las conversaciones. Escucha a tu interlocutor y elige las conversaciones que quieres que Saúl tenga con él. Elige qué decirle mediante el menú de opciones de conversación que aparece en la pantalla al hablar.

5. **Conceptos básicos de manejo del juego**

   Antes de emprender la aventura debes identificarte con un nombre de usuario y una dirección de correo electrónico que quedarán asociados a todos los progresos de la partida.

   5.1 Identificación de personajes y objetos

   En la pantalla de juego encontrarás distintos personajes y objetos. Para interactuar con ellos es necesario realizar las acciones de mirar, coger, usar o hablar.

   5.2 Acciones

   Las acciones que puedes realizar dentro de una escena son las siguientes:

   - **Siguiente pantalla**
     El botón “Siguiente pantalla” habilitado indica que tienes que dirigirte a un escenario distinto, en el que encontrarás nuevos objetos o personajes con los que poder relacionarte. Haz *click* en el botón y Saúl se desplazará a ese lugar.
   - **Mirar/Coger/Usar**
     Existen varias acciones sobre los objetos que aparecen en la escena. Muchos de ellos encierran información necesaria para progresar en el juego y es necesario cogerlos para usarlos en escenas posteriores. También es posible examinar objetos dentro del inventario, es decir, objetos que has decidido recoger para utilizarlos posteriormente. Solo es necesario seleccionarlos y mirarlos para obtener su descripción.

5.3 El inventario

El inventario está formado por todos los objetos que recoges a lo largo del juego y de los que puedes servirte para avanzar en la aventura.

5.3.1 Cómo usar los objetos

Para usar dos objetos es necesario haberlos examinado antes y pulsar el botón “Usar”. Aparecerá un desplegable con todos los objetos almacenados en el inventario en el que hay que elegir el primer objeto. Una vez elegido aparecerá otro desplegable con el resto de objetos, los que están almacenados en el inventario y los que están en la escena actual. Por tanto se pueden usar dos objetos si:

- Los dos objetos están almacenados en el inventario.
- Un objeto está en el inventario y el otro en la escena en la que te encuentras en ese momento.

En el caso en el que dos objetos no pueden usarse, Saúl se negará a realizar la acción negándose de diferentes formas.

6. La ayuda de Saúl

Saúl se moverá por los distintos escenarios enviándote a tu correo electrónico la información del lugar donde se encuentra, la información que va consiguiendo y te pedirá ayuda en ciertas circunstancias. Para recibir las pistas de Saúl, es necesario estar conectado a internet.

7. Personajes

- Saúl

  Joven enamorado de su amiga Lidia. Sufre un ligero tartamudeo al hablar con ella. Se le dan mal las matemáticas desde pequeño y le sigue costando aprobarlas. Además los juegos de lógica no son su fuerte.

- Lidia

  Hija de un importante cargo de la policía y un poco mimada. Es una chica extrovertida a la que le gusta divertirse y montar fiestas.

- Gran Maestre

  Una persona brillante que emplea su inteligencia para convertirse en un millonario.

- Jugador

  Persona bastante maja, agradable e inteligente (o eso se cree) al que se le dan bien los problemas de lógica que ayuda a Saúl a lo largo de toda la aventura.

BIBLIOGRAFÍA

LIBROS

[1] CORREA URIBE, Guillermo. *Desarrollo de Algoritmos y sus aplicaciones*, III Edición. USA: MacGraw - Hill Inc, 1992. 958-600-109-1.

[2] TYLER, Jenny. HOWARTH, Les. *Programa tus propias aventuras en tu computadora*, I Edición. Madrid: Ediciones PLESA, 1985. 84-7374-137-4.

[3] BOOCH, Grady. RUMBAUGH, James. JACOBSON, Ivar. *El lenguaje unificado de modelado,* I Edición. Madrid: Addison Wesley Iberoamericana, 1999. 84-7829-028-1.

DOCUMENTOS

[4] MONTOYA, Carlos. FLORES, Pablo. Los puzles en alambre como recursos didácticos para la enseñanza de las matemáticas. *Puzzles de alambre.* [en línea]. 2003 [ref. de enero de 2011]. Disponible en Internet: http://www.ugr.es/~pflores/textos/aRTICULOS/Propuestas/Articulo_Gaceta_Montoya_Flores.pdf

[5] NAVARRO, David. ”Los videojuegos de puzles mejoran nuestro cerebro. El Mundo”. [en línea]. *El Mundo.* 21 octubre 2010. http://www.elmundo.es/elmundo/2010/10/21/navegante/1287659301.html [consulta: enero 2011].

[6] VILLANUEVA ROSA, Sara. *Juegos de Realidad Alternativa: ARG.* Universidad Carlos III de Madrid, Departamento de Telemática, 2008.

[7] *An alternate reality game for language learning: ARGuing for multilingual motivation.* [en línea]. Oxford (Reino Unido): University of the West of Scothland, School of Computing, 2011-[ref. de marzo de 2011]. Disponible en Internet: < http://www.elsevier.com/wps/find/journaldescription.cws_home/347/description>. ISSN 0360-1315.

[8] LEVITIN, Anany. PAPALASKARI, Mary-Angela. *Using Puzzles in Teaching Algorithms* [en línea]. USA: Villanova University, 2002-[ref de enero de 2011]. Disponible en Internet:< http://www.sigcse.org. >ISSN: 0097-8418.

[9] GARCÍA FERNÁNDEZ, Fernando. ”Nativos interactivos: Los adolescentes y sus pantallas: reflexiones educativas.”. En: *V Congreso Internacional de EducaRed*, (Madrid 26-28 de noviembre de 2009). V.I. Madrid: Foro Gerneraciones Interactivas, 2009. ISSN: 1681-5653.

[10] SHARDA, Nalin K. *Designing, Using and Evaluating Educational Games: Challenges, Some Solutions and Future Research* [en línea]. Melbourne City MC, Australia: School of Computer Science and Mathematics Victoria University, 2008- [ref de enero de 2011]. Disponible en Internet: < http://www.mendeley.com/research/designing-using-and-evaluating-educational-games-challenges-some-solutions-and-future-research/>.

DIRECCIONES WEB

[11] ELOTROLADO. *Historia de los videojuegos: Los inicios.* [en línea] <www.elotrolado.net/wiki/Historia_de_los_videojuegos:_Los_inicios#Los_inicios_de_los_videojuegos> [Consulta: enero 2011].

[12] WIKIPEDIA. *Características de los videojuegos de realidad alternativa*. [en línea] <http://en.wikipedia.org/wiki/Alternative_reality_game> [Consulta: enero 2011].

[13] BLOG AVANT GAME. *A blog about why games make us happy and how they can change the world*. < http://blog.avantgame.com/> [Consulta: enero 2011].

[14] JANE McGONIGAL. *Games*. http://janemcgonigal.com/play-me/. [Consulta: agosto 2011].

[15] TED. *Ideas worth spreading.* <http://www.ted.com/> [Consulta: enero 2011]

[16] WIKIPEDIA. *Perplex City*. <http://en.wikipedia.org/wiki/Perplex_City>. [Consulta: enero 2011]

[17] WIKIPEDIA. *The Lost Ring*. <http://www.thelostring.com/>[Consulta: enero 2011]

[18] TOP SECRET. *Top Secret Dance Off*. <http://topsecret.ning.com/> [Consulta: enero 2011]

[19] CRYPTOZOO. *Cryptozoo.* <http://cryptozoo.ning.com/> [Consulta: enero 2011]

[20] SUPERSTRUCT GAME. *Explore the World of Superstruct.* http://www.iftf.org/SuperstructGame. [Consulta: enero 2011]

[21] ZONA EL INTERNADO. *¿Dónde está Yago?* <http://www.zonaelinternado.com/donde-esta-yago-ayudanos-a-encontrarlo/>. [Consulta: enero 2011].

[22] THE LOST EXPERIENCE. *The lost experience begins!* www.thelostexperience.com/. [Consulta: enero 2011].

[23] LOSTPEDIA. *The Lost Experience*. <http://lostpedia.wikia.com/wiki/The_Lost_Experience>. [Consulta: enero 2011].

[24] BUSINESSWEEK. *Audi's New Viral Campaign is Catchy.* <http://www.businessweek.com/the_thread/brandnewday/archives/2005/05/audi_has_finall.html.> [Consulta enero 2011].

[25] WIKIPEDIA. *The Beast.*<http://en.wikipedia.org/wiki/The_Beast_(game)> [Consulta: enero 2011]

[26] TINTADIGITAL. *La primera víctima de Batman fue la red.* http://www.tintadigital.org/2008/08/18/la-primera-victima-de-batman-fuela-red/. [Consulta: diciembre 2009]

[27] YOUTUBE. Secuenstro de Lucas Montano. http://www.youtube.com/watch?v=Db_YexR_hPE. [Consulta: enero 2011]

[28] IDG. *Ya está en marcha Suceso Tierra*. <http://www.idg.es/pcworld/Ya-esta-en-marcha-Suceso-Tierra_-un-juegogratuito/doc54007-Juego.htm>. [Consulta: enero 2011].

[29] I LOVE BEES. I love bees. <www.Ilovebees.com> [Consulta: enero 2011].

[30] WORLD WITHOUT OIL. Lessons plans. <http://worldwithoutoil.org/metateachers.htm>. [Consulta: enero 2011].

[31] SERIOUS GAMES MARKET. *New ARG EVOKE: Serious games changing the world.* <http://seriousgamesmarket.blogspot.com/2010/01/new-arg-evoke-serious-games-changing.html >. [Consulta: enero 2011].

[32] FLOATLEARNING. *Alternative Reality Games(ARGs) as Mobile Learning.* http://floatlearning.com/2011/02/alternative-reality-games-args-as-mobile-learning/ [Consulta: enero 2011].

PRESUPUESTO

Todo proyecto siempre tiene un apartado de presupuesto en el que se debe analizar el tiempo dedicado a él, los medios utilizados, los costes que le han producido al desarrollador, y luego aplicarle el margen de beneficio que se estime oportuno que se quiere ganar en el proyecto realizado.

Para conocer el esfuerzo del diseñador del proyecto, además de la etapa de programación hay que considerar ciertas tareas que también fue necesario realizar, y que se han considerado para calcular su duración:

- Estudios de antecedentes y documentación: Consiste en buscar información relacionada con el proyecto, antecedentes, búsqueda de documentación útil, etc. Con esta tarea se comienza el proyecto, pero es una actividad continua durante todo el desarrollo de éste.

- Recordatorio y aprendizaje del lenguaje de programación orientado a objetos: El estudio y aprendizaje de la programación orientada a objetos es una de las materias que se imparte principalmente en los primeros años de carrera, por lo que ha sido necesario un refresco de esta materia y aprender a usar nuevas bibliotecas de Java.

- Diseño de la aplicación: Capas necesarias que debe tener la aplicación y diseño y funcionalidad de cada una de esas capas.

- Implementación de la aplicación: La implementación de la aplicación ha sido la principal tarea de este proyecto. En esta fase se incluyen la portada y presentación del videojuego, código, creación de la base de datos de la partida, incluyendo los diálogos con los personajes, las pruebas lógicas que el jugador debe superar, y la redacción de los emails.

- Fase de Pruebas: Toda aplicación debe ser probada para ver su robustez y conocer si gusta o no por medio de los probadores de videojuegos.

- Redacción de la memoria: La memoria del proyecto se ha realizado una vez finalizado el proyecto. Con el fin de tener la visión completa y definitiva del trabajo realizado.

Para calcular el coste total de este trabajo, se va a desglosar el presupuesto según los costes ocasionados:

a) Costes personales

En este caso el diseñador y programador del videojuego es el proyectando. Normalmente una aplicación de estas características está compuesta por un amplio grupo de personas, ya que un videojuego suele estar formado por guionistas, programadores, desarrolladores gráficos, etc. En este caso, todas estas tareas las ha tenido que realizar el proyectando.

Apellidos y nombre	Categoría	Dedicación (hombres mes)	Coste hombre mes	Coste (Euro)	
Carmen Martínez Ferrari	Ingeniero	12	2.694,39	32.332,68	
			Total	32.332,68	€

b) Costes equipos

El equipo utilizado en este trabajo ha sido el ordenador personal del proyectando, cuyo coste se detalla a continuación:

Descripción	Coste (Euro)	%Uso dedicado proyecto	Dedicación (meses)	Periodo de depreciación	Coste imputable
Intel Core Dúo	500,00	100	12	60	100,00

	Total	100,00 €

- Intel Core Dúo: Para poder realizar la programación del presente proyecto y la generación de la memoria correspondiente.

c) Otros costes

Otros costes relacionados con la creación del proyecto son los siguientes:

- Microsoft Office 2007: El Office es necesario para poder realizar el documento final de la memoria del proyecto y dibujar los diagramas UML realizados para la memoria de este proyecto. También se ha usado el programa Power Point para modificar algunas imágines del juego.

- Microsoft Visio: Utilizado para pintar diagramas UML necesarios para la memoria del proyecto.

- Editor de programación en Java: Editor de programación en Java: Se ha necesitado un editor de Java, para simplificar el proceso de programación de la aplicación, ya que los editores actuales permiten ver las líneas, identifican palabras claves, cierran paréntesis, llaves, etc. Esta aplicación también ha sido gratuita ya que hay cientos de aplicaciones gratuitas que permiten ya esas funcionalidades.

- Kit Desarrollo Java: En este caso, el kit de desarrollo para programar en Java no ha proporcionado ningún tipo de coste ya que puede ser descargado gratuitamente de la página oficial de ORACLE.

- Tarifa ADSL: Una conexión a Internet ha sido necesaria para poder descargar las últimas versiones de Java de la página oficial de ORACLE, para obtener información de documentación sobre tecnologías y para la realización del documento final, para obtener programas para editar código fuente en Java y en XML, etc. El coste de la tarifa es de 36 € y se ha utilizado durante 10 meses.

Descripción	Coste imputable
Microsoft Office 2007	140,00
Editor de programación en Java	0,00
Microsoft Visio	0,00
Kit desarrollo Java	0,00
Tarifa ADSL	360,00
Total	500,00 €

d) Resumen de los costes

Para calcular los costes totales se han tenido en cuenta una tasa de costes indirectos del 2,5%.

Presupuestos Costes Totales	Presupuestos Costes
Personal	32.332,68
Amortización	100
Subcontratación de tareas	0
Costes de funcionamiento	500

Costes Indirectos	823,3
Total	33.756 €